commodore Magazine

AÑO III · Núm. 27
Mayo 1986 300 Ptas.

REVISTA INDEPENDIENTE PARA USUARIOS

SUPLEMENTO
PROGRAMAS
12 PAGINAS

C-128 Comandos de disco

Sistemas de numeración

Ficheros secuenciales

INTELIGENCIA ARTIFICIAL

Director:
Rubén Sanz
Coordinador Editorial:
J. Ignacio Rey
Redacción:
Teresa Aranda
Colaboradores:
José D. Arias
Alejandro de Mora-Losana
Paloma Saco
Diseño:
Benito Gil
Edita
PUBLINFORMATICA
Bravo Murillo, 377 - 5.° A
Telf.: 733 74 13. Madrid - 28020
Presidente:
Fernando Bolín
Director Editorial
Revistas Usuarios:
Juan Arencibia
Director de ventas:
Antonio González
Jefe de Producción:
Miguel Onieva
Servicio al cliente:
Julia González - Telf.: 733 79 69
Publicidad:
Emilio García
Dirección, Redacción y Publicidad:
Bravo Murillo, 377 - 5.° A
Telf.: 733 74 13
Publicidad Barcelona:
María del Carmen Ríos
Pelayo, 12
Telf.: (93) 301 47 00
ext. 27-28 y (93) 318 02 89
08001 BARCELONA
Depósito Legal: M-6622-1984
Distribuye: S.G.E.L.
Avda. Valdelaparra, s/n.
Alcobendas. Madrid
Distribuidor en Venezuela:
SIPAM, S. A.
Avda. República Dominicana
Edif. FELTREE
Boleita Sur Caracas (Venezuela)
Distribuidor en Argentina:
DISA
Sta. Magdalena, n.° 541
Buenos Aires (Argentina)
Fotocomposición: Consulgraf
Nicolás Morales, 34. 28019 Madrid
Fotomecánica: Karmat
Pantoja, 10. Madrid
Imprime: G. Velasco, S. A.
Solicitado control O.J.D.
Esta publicación es miembro
de la Asociación de Revistas
de la Información ARI, asociada
a la Federación Internacional
de Prensa Periódica FIPP.

El P.V.P. para Ceuta, Melilla y
Canarias, incluido servicio
aéreo, es de 300 ptas. sin IVA

Rogamos dirijan toda la correspondencia relacionada con suscripciones a Commodore Magazine. Edisa. Tel.: 415 97 12. López de Hoyos, 141-5. 28002 Madrid. Para todos los pagos reseñar solamente Commodore Magazine.
Para la compra de ejemplares atrasados dirigirse a la propia editorial Commodore Magazine, Bravo Murillo, 377 5° A. Tel.: 733 74 13. 28020 Madrid.

Editorial

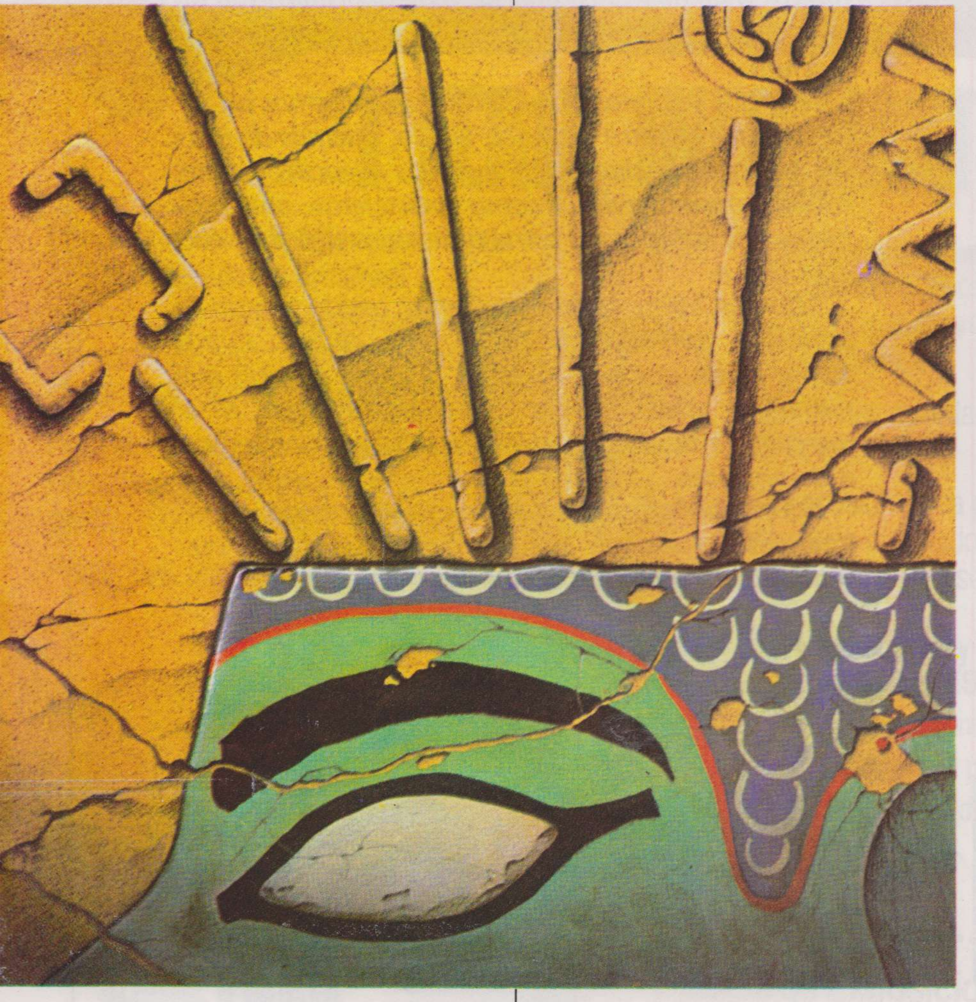
Uno de los caminos por los cuales avanza el mundo de la Informática, es el que busca el desarrollo de la Inteligencia Artificial: el intento de simular los razonamientos lógicos del hombre por medio de máquinas; la programación de impulsos eléctricos capaces de albergar en memoria deducciones propias en función a unos primeros datos; reducir el tamaño de las unidades de memoria, con la posibilidad de acumular información y ampliar el campo comparativo; nuevos materiales que aplican más velocidad, son algunas de las primeras metas que se están consiguiendo.

Hoy nos sentimos obligados a replantearnos un futuro donde nuestro pequeño aliado, además de ejercer un control exhausto de nuestros negocios o necesidades, puede servirnos como «consejero», ante problemas de complicada estructuración o difícil resolución lógica.

Mientras tanto, todavía no hemos tenido tiempo de asimilar y aprender nuevas técnicas, cuando somos sorprendidos por los resultados de otros avances que nos dejan atrás. Nuevos fabricantes avanzan en el mercado, a la vez que otros consolidan sus nombres o desaparecen.

Porque la Informática sigue evolucionando, aventurémonos a deducir que nos depara un futuro cercano que todos vamos a conocer.

Sumar





rio







NOTICIAS

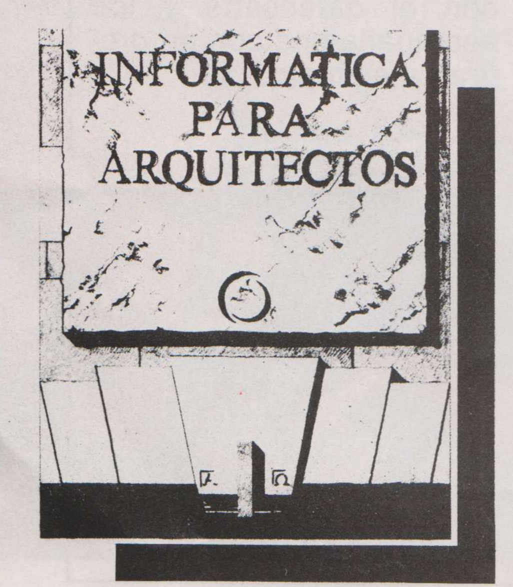
I Concurso de carteles Soft/Hewlett-Packard, Informática para arquitectos

El motivo de esta convocatoria es, principalmente, introducir a las jóvenes promociones y estudiantes de arquitectura, en el campo de la microinformática profesional, herramienta de primera necesidad en cualquier gabinete técnico.

En él podrán participar todos los arquitectos superiores españoles titulados entre el 1 de enero de 1979 y el 31 de Diciembre de 1985, y los estudiantes matriculados en Proyecto Fin de Carrera y/o 6° curso de Arquitectura, en cualquier Escuela Técnica Superior del país.

El premio único del concurso, valorado en 1.800.000 ptas., está compuesto por la biblioteca de programas SOFT, específicos para el mundo de la arquitectura, junto con un ordenador Hewlett-Packard 150 II y una impresora gráfica Think Jet de chorro de tinta.

En este concurso colaboran la entidad GRUPCAIXA, en cuyas oficinas de la Castellana, 51, de Madrid, se celebrará la exposición de los carteles y entrega del premio.



Serie de aventuras y misterio para ordenador.

IDEALOGIC, S.A. emprende el camino de los juegos de aventura con el lanzamiento del programa Profesión Detective para el Commodore 64, arropado de un fabuloso concurso con 56 premios a los que optarán todos los usuarios mayores de 12 años.

Con este programa se ha creado el concurso «Busca el misterio», para todos aquellos poseedores del programa deberán contestar a las siguientes preguntas:

a) Nombre del culpable
b) ¿Cómo lo hizo?
c) ¿Por qué?

Las bases de participación van adjuntas a cada uno de los programas. La fecha tope para recibir las soluciones será el próximo día 15 de julio.

El reparto de premios se efectuará en los comercios donde se hayan vendido los programas de los ganadores, siendo el primer premio de 100.000 pesetas.

Commodore PC 10-II y PC 20-II

Commodore, a partir del mes de abril, comercializará en España la nueva serie de su conocido Commodore PC en sus versiones PC 10-II y PC 20-II, sin que el incremento de características que aportan suponga ningún aumento de los precios, que seguirán siendo de 353.000 ptas. para el PC 10-II, y 520.000 ptas., el PC 20-II.

La nueva configuración stándar que presentará el Commodore PC es la siguiente:

512 Kb de memoria mínima de RAM.
Adaptador Gráfico Avanzado para color y monocromo.
BIOS compatible XT.
Disco fijo de 20 Mb (PC 20)
Teclado y Sistema Operativo (MS-DOS) en castellano.



Amstrad compra a su rival Sinclair

Amstrad ha comprado a su máximo rival y pionero en el sector, la también británica Sinclair, por 5 millones de libras, unos 1.150 millones de pesetas.

Según los términos del acuerdo que se conocen actualmente, Amstrad ha adquirido los derechos de fabricación y venta en el mundo entero de todos los ordenadores y productos actuales y futuros de Sinclair Research.

Clive Sinclair reconoció al hacerse público el acuerdo que «nosotros fuimos los innovadores, pero sabíamos que no podríamos competir con los expertos en marketing». Alan Michel Sugar Trading (cuyas siglas forman el nombre de su empresa), comenzó sus actividades vendiendo películas a sus compañeros de colegio, para introducirse después en el negocio de la venta de antenas para radios de coche, estuches para discos, material de alta fidelidad y vídeo. Sugar creó Amstrad de la nada y se basa su éxito en la reventa de ordenadores simples y económicos fabricados en Corea del Sur.

Red informática norteamericana para simulación de batallas espaciales

Las instalaciones consistirán en una red de ordenadores basados en laboratorios gubernamentales e industriales, además de grupos de misiles. Serán construidos como parte de la Iniciativa de Defensa Estratégica (Guerra de las Galaxias), que con un coste de 26.000 millones de dólares ha sido impulsada por el presidente Reagan. Su objetivo está dirigido a la formación de una estrategia orientada a defender el Occidente contra un ataque nuclear permitiendo derribar misiles en pleno vuelo.





La Inteligencia Artificial supone el intento del hombre de crear algo cuyos resultados sean similares a los del pensamiento humano, de forma que desde el exterior los fines alcanzados sean los mismos que los que podríamos haber conseguido nosotros. LISP, LOGO y más apropiadamente el PROLOG son lenguajes que se intentan acercar a los razonamientos lógicos

Necesito ayuda, es verdad.
—¿Qué representaría para usted conseguir una ayuda?
Tal vez pudiera entenderme con mi madre.
—Digame algo más de su familia.
Mi madre cuida de mí.
—¿Quién más de su familia cuida de usted?
Mi padre.
—Su padre...
Usted se parece a mi padre en ciertas cosas.
—¿Qué parecido nos encuentra?
Usted no es muy agresivo, pero creo que quiere que yo no lo note.
—¿Qué le hace pensar que no soy muy agresivo?
No discute conmigo.
—¿Por qué cree que no discuto con usted?
Me tiene miedo.
—¿Le agrada creer que le tengo miedo?





Inteligencia Artificial

Este chocante diálogo, que parece sonsacado de la consulta de algún psicoterapeuta, no es sino el resultado de una conversación entre una chica y un frío y lógico terminal de ordenador.

El programa, llamado Eliza en honor a la Eliza de «Pigmalión», que podía aprender a hablar cada vez mejor, es obra de un profesor del Instituto de Massachusetts llamado Joseph Weizenbaun, y se hizo extraordinariamente famoso, hasta el punto de que se llegó a pensar en «una red de terminales psicoterapeutas de ordenadores, a semejanza de grandes cabinas telefónicas, en las cuales, por unos cuantos dólares la sesión, podremos conversar con un atento, capacitado e independiente terapeuta», como dio a entender Carl Sagan, en «Natural History».

Incluso la secretaria de Weinzenbaun en una ocasión en que se dispuso a hablar con Eliza, rogó al profesor que abandonara la estancia. ¡Hasta tal punto veía a la máquina como confidente!.

Curiosamente, Weizembaun, quien apoyó con su programa a muchos de los seguidores de la llamada «Inteligencia Artificial», fue posteriormente uno de los detractores más serios y coherentes que ha tenido ésta, y su objetivo fue demostrar la clara diferencia entre un ordenador y una mente humana, y el peligro que podía representar el primero si no se tomaban las medidas oportunas.

¿QUE ES LA INTELIGENCIA ARTIFICIAL?

La IA, como ya se la llama en casi todos los sitios, es el intento del hombre de crear algo cuyos resultados sean similares a los del pensamiento humano, ya sea simulando nuestro intelecto y nuestra forma de pensar o bien por otros medios, de forma que desde el exterior los fines alcanzados sean los mismos que los que podríamos haber conseguido nosotros. (Y si pueden ser, mejores).

Esto último significa que hay dos formas de entender cómo conseguir que un ordenador «piense». La primera es creando un programa que discurra y piense lo más parecido posible a nosotros, con nuestros defectos y nuestras virtudes y por lo tanto las conclusiones sean semejantes a las nuestras, (si acaso, sólo diferentes en velocidad.)

La segunda prescinde de

cómo lo hagamos nosotros; lo único que interesa es el resultado. Si lo que se pretende es que juegue al ajedrez, se ha de construir un programa que lo haga, de la forma que sea, pero que al final el resultado sea un movimiento de una pieza. Ya veremos más tarde qué significan ambas visiones. Ahora vamos a remontarnos a la Inglaterra de principios de siglo.

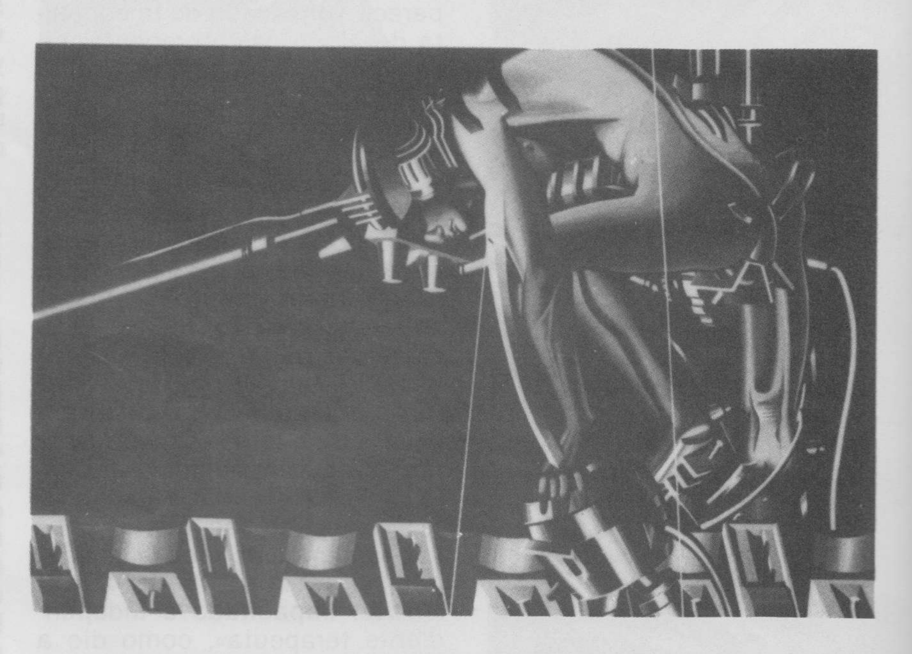
En aquellos años nace uno de los matemáticos más importante de este siglo, y una de las mentes mejor dotadas que se conocen: Alan Turing. En honor a la verdad, no se puede decir que Turing estuviera muy en su sano juicio. Por ejemplo, invento una variante del ajedrez en la que el jugador que dando una vuelta a la manzana llegara antes del movimiento del otro tenía derecho a una jugada adicional. (Por supuesto, Turing fue campeón universitario en las carreras, de forma que la mayoría de las veces conseguía esa jugada, amén de poner nervioso al contrario), y, para colmo de males, se suicidó con cianuro cuando todavía tenía 41 años. Pero aparte de sus excentricidades y extrañezas, sus valiosísimos estudios, y su ya famosa «Máquina de Turing» (un ordenador que no tenía limitaciones en su memoria y que sólo existe en teoría, pero no se puede llevar a la práctica), le hicieron ser uno de los principales padres de la Inteligencia Artificial. Profetizó con admirable intuición lo que sería el día de mañana la informática y la ingeniería del conocimiento, y creó una curiosa prueba para probar si una máquina es inteligente o no. Se le ha llamado el test de Turing, y consiste aproximadamente en lo siguiente:

Se sitúa una persona delante de dos terminales, y ésta le va haciendo una serie de preguntas a cada uno de ellos; a un lado de uno de los terminales hay una persona, y al otro, un programa de ordenador. Si la persona que hace las preguntas es incapaz de dilucidar quién es la persona y quién el programa «engañador», ese programa, y por tanto la máquina en ese momento, son inteligentes. Lo único que implica en realidad es que «una cosa piensa si lo parece». Eso, que parece una tontería, no lo es tanto, si uno mira otras y observa la cantidad de discusiones y críticas que han tenido frases como esta.

Aunque al principio nadie creyó en la Inteligencia Artificial, poco a poco algunos avances empezaron a vislumbrar lo que podía ser el cambio más importante de la historia de la humanidad. Uno de esos avances fue un programa creado por Newel y Simon en 1957 que demostraba teoremas de lógica matemática de una forma similar a como lo hacía un matemático normal. Como anécdota, podemos contar que estos dos autores llegaron a la conclusión de que algunos teoremas tardarían muchos miles de años en demostrarse por ordenador: al año siguiente un japonés hizo un programa similar que tardó un cuarto de segundo en hacerlo. Hubo más intentos, y se vaticinó que en unos pocos años se tendría un traductor de idiomas, pero no fue así. Los problemas que aparecieron cuando se intentó entender el lenguaje, dieron lugar a que incluso los más optimistas vieran ya lejos el día en que el jefe de su empresa fuera un IBM. Estos problemas podríamos imaginarlos así:

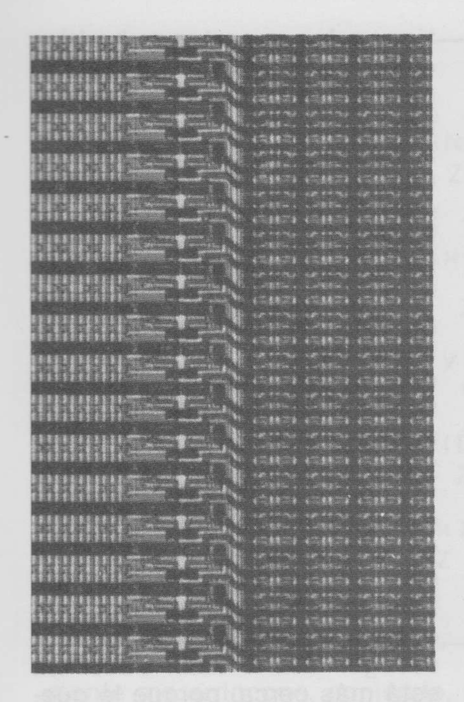
Supongamos que alguien nos dice «Se me cayó el plato, dio con el pico de la mesa y se rompió».

Todos entenderíamos sin lugar a dudas que, evidentemente, fue el plato el que se rompió, ya que a duras penas puede un



plato romper una mesa. Pero la respuesta no está tan clara para el ordenador. De hecho, si la frase hubiera sido:

«... dio contra la figura de porcelana y se rompió», nosotros mismos habríamos vacilado. En el primer caso no dudamos porque sabíamos lo que era una mesa, y sabíamos que una mesa es mucho más dura que un plato. Es decir, NOSOTROS TENIAMOS CONOCIMIENTOS



DE LA REALIDAD. Para implementar esto en el ordenador, tendríamos que decirle cómo es una mesa, sus características, incluida su dureza, su pesadez, etc... y además, deberíamos decirle cómo es una mesa cualquiera, no una particular. Una mesa puede tener tres patas o ser de mármol, pero un perro sólo puede tener cuatro, y eso ha de comprenderlo. Así que, ante este problema, las investigaciones se han dirigido en el sentido de introducir unos pocos conocimientos sobre alguna materia determinada y hacer que el ordenador «piense» sobre esos pocos datos. Los «sistemas expertos» son algo parecido.

No obstante, sí que se han conseguido algunos logros espectaculares. Uno de los más sonoros fue el programa BKG 9.8 de Hans Berliner, que en julio de 1979 consiguió derrotar al campeón del mundo de Backgammon, en Monte Carlo, con un premio de 5.000 dólares, y después de siete partidas, le ganó con un rotundo 6-1.

LA IA Y LOS LENGUAJES DE PROGRAMACION

Hay muchas formas de crear algo que parezca inteligente, y no ha de ser forzosamente en ningún lenguaje especial. Podemos incluso hacerlos en Basic, y el resultado puede ser muy similar, aunque seguramente más lento. Si lo que nosotros queremos es un programa que juegue con nosotros a algo, cosa que requeriría de alguna forma «pensar», podemos crear el programa si el juego tiene solución y la conocemos. Hay muchos juegos que lo cumplen, como los juegos con palillos o con cerillas (Nim), con los que puedes impresionar a las visitas que intenten jugar contra el ordenador. Un típico juego que se suele meter en las calculadoras, y que vosotros podeis programar, es el siguiente:

— Se tienen objetos (podeis hacerlo con naranjas, cubos, pingüinos...) y dos jugadores.

— Se escoge un valor máximo de objetos, digamos p, que se podrán coger como máximo en cada jugada.

— Una vez cada uno, los jugadores van tomando 1, 2... hasta p elementos de una sola vez (siempre se tienen que llevar alguno), y aquel que se lleve el último será el perdedor.

Supongamos que tenemos 21 balas y un fusil que tira 1, 2 o tres balas. En este caso particular n sería 21 y p sería 3.

El jugador A podría tirar por ejemplo 2. Quedarían 19. El B tira 3. Quedarían 16.

El juego continuaría hasta que quedara uno sólo. Aquél que tire el último, habrá perdido.

Si observais el desarrollo del juego, tiene truco. Si el primero mueve:

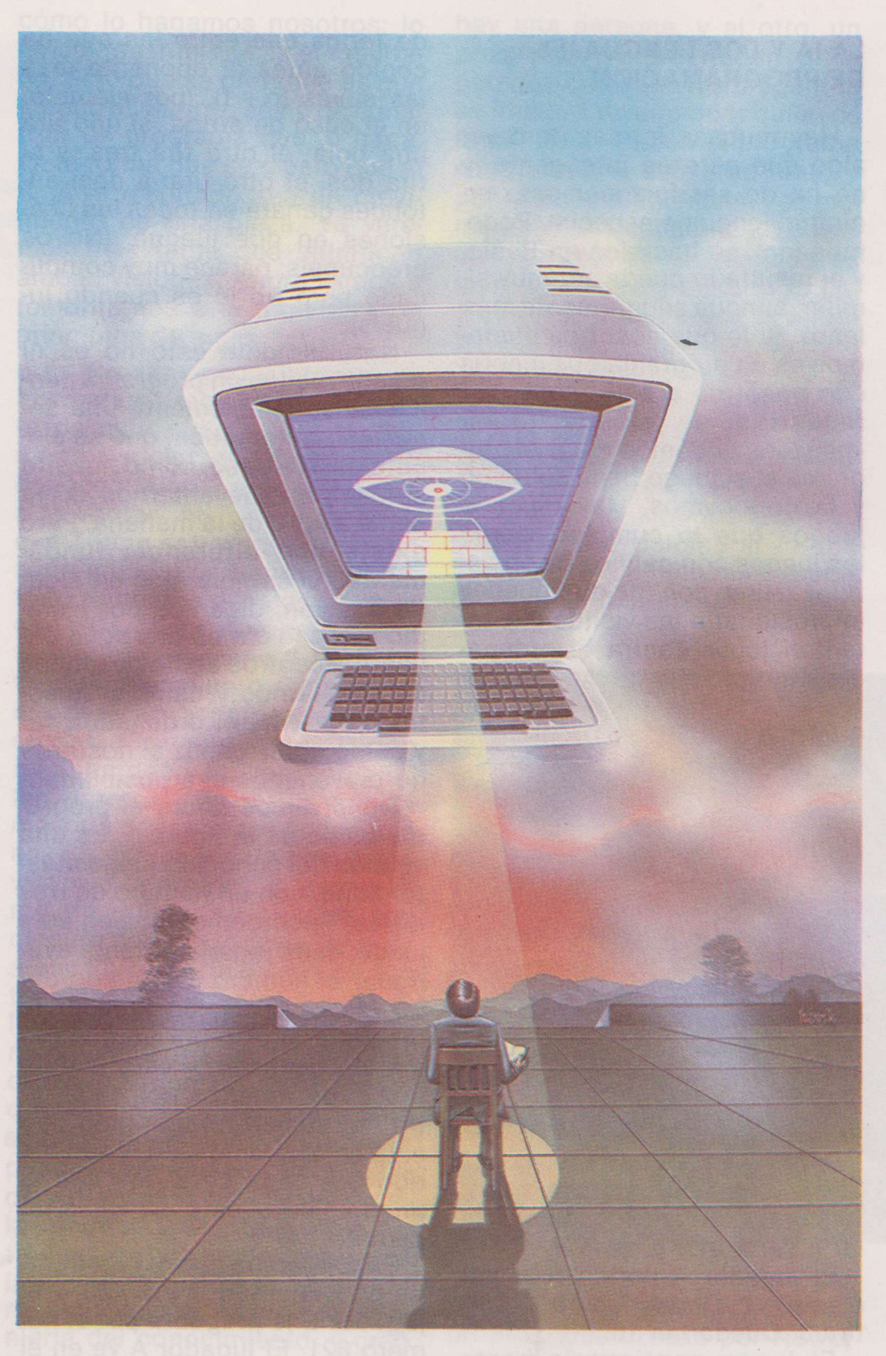
$$(n\text{-}p\text{-}2) - [(\mathrm{INT}\frac{(n\text{-}p\text{-}2)}{(p+1)}) \times (p+1)]$$

y a partir de ahí mueve siempre de forma que entre las que ha cogido antes el oponente más las suyas (p + 1), (por ejemplo, en el caso de arriba, si uno tira una bola, el otro tira tres, y si tira dos, el otro tiraría dos) entonces ganará en todas las ocasiones en que juegue. (No os preocupeis; parece muy complicado pero no lo es cuando jugueis).

Naturalmente esto no es ni mucho menos un programa pensante, es simplemente una secuencia automática, que se ejecuta paso a paso, como nuestra actuación al levantarnos de la cama a las 6 de la mañana. Pero existe una diferencia fundamental: si cambia algo en alguna ocasión, digamos por ejemplo que ese día no hay agua caliente, nosotros somos capaces de adaptarnos y cambiar nuestro programa; el ordenador, tal y como no es capaz si nosotros no prevenimos esa posibilidad.

Observemos ahora el programa basic de la figura 1. Es una versión del conocido juego Master Mind, con un número de tres cifras. Recordemos que en este juego, cada jugador intenta averiguar el número escondido del otro, y para eso un jugador le pregunta al otro por un número que ha pensado, y éste le contesta «un muerto» por cada cifra del número preguntado que esté en el mismo sitio, y «un herido por cada cifra que sea la misma, pero esté en un sitio distinto.

Supongamos que el jugador A ha pensado el número 426, y el jugador B le pregunta por el número 621. El jugador A ve en el número escondido que el 2 está en el mismo lugar, y que el 6 está, aunque en un sitio distinto. Serían entonces 1 muerto y 1 herido. Por supuesto, si B le hubiera dicho 426, serían 3 muertos, y A hubiera perdido, mientras que si hubiera dicho 264 serían 3 heridos, ya que son todos los números pero están cambiados de sitio. En el caso



del programa de la figura, uno de los jugadores es nuestro querido Commodore, y el otro sois vosotros. Y os aseguro una cosa: si le dejais empezar a él y jugais 10 partidas... ¡Seguro que os gana más de 6!

Y ¿Cómo lo hace? Este programa no es tampoco verdaderamente «pensante», y, aunque se acerca más a la verdadera inteligencia artificial que el anterior de las balas, lo único que aprovecha son las salvajes dotes de cálculo del procesador, y lo hace de la siguiente forma:

El nos pregunta por un número. Nosotros ya lo hemos pensado antes, y podemos decirle cuántos muertos y heridos hay. El entonces calcula y almacena todos los posibles números que tienen los muertos y heridos que le hemos dicho, y lo guarda en una matriz. El número desconocido que hemos pensado nosotros está en esa matriz, es uno de ellos, ya que tiene que cumplir lo que hemos dicho. En la vez siguiente, después de haber jugado nosotros, él nos vuelve a preguntar por un número, esta vez uno cualquiera de los almacenados antes. Ese podría ser el nuestro, y, si no lo es, vuelve a guardar todos los de la matriz que cumplen los muertos y heridos que le hemos especificado. El resultado es que cada vez le quedan menos, hasta que ya sólo le quede uno, en cuyo caso esa es la solución. En realidad lo que hace es calcular todas las posibles soluciones y escoger una al azar. Cada vez está más cerca, porque le quedan menos posibilidades donde escoger.

Fácil, ¿no? Pero esto es puro cálculo bruto, y no un verdadero programa inteligente. Puede ocurrir que el número de posibilidades total para guardar sea tan grande que ni el ordenador más potente de la NASA sería capaz de calcularlas en miles de millones de años. Este es el caso del ajedrez que en su programación tiene que cambiar la táctica y... ¿Cuál escoger?

La única que podemos seguir es intentar que el ordenador siga los mismos pasos que seguimos nosotros.

Suponeros que estamos jugando una partida de ajedrez, con nuestro ya conocido jugador B. En ese momento él, contra todo pronóstico, mueve el caballo y nos da jaque. Las ideas que corren por nuestra cabeza son algo así:

— si muevo el rey allí, él me volverá a dar jaque con la torre, pero entonces yo podría taparlo con la dama. Claro que el me haría entonces jaque doble. No, mejor no».

— Si me como su caballo, él lo estará esperando y segura-

```
((DERI "X" 1))
((DERI X 0)(NUM X))
((DERI (X | Y) Z)(EQ Y (+ | x))

        (DERI X y)(DERI x z)
        (EQ Z (x+y)))

((DERI (x | y) Z)(EQ  Y (* | x))

      (DERI X y)(DERI x z)
      (EQ Z (y * x + X * z)))

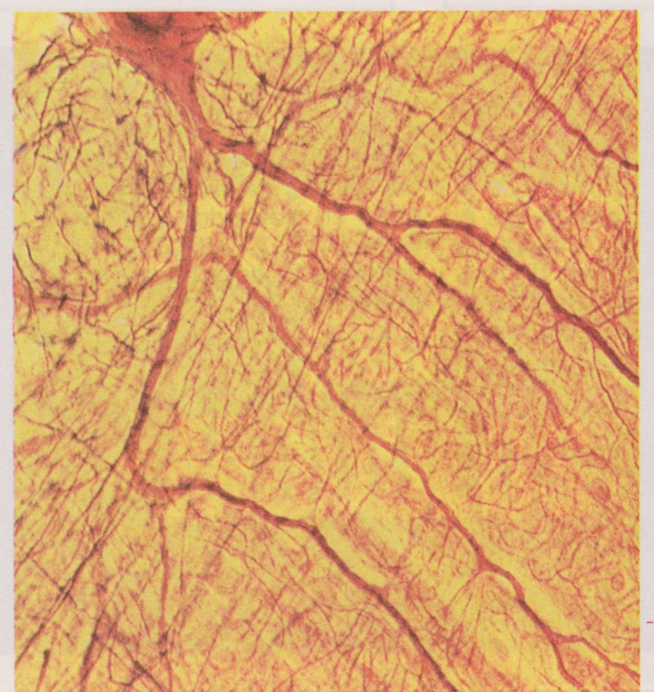
((DERI (X) Y)(DERI X Y))
((DERI (X | Y) Z)(EQ X SEN)

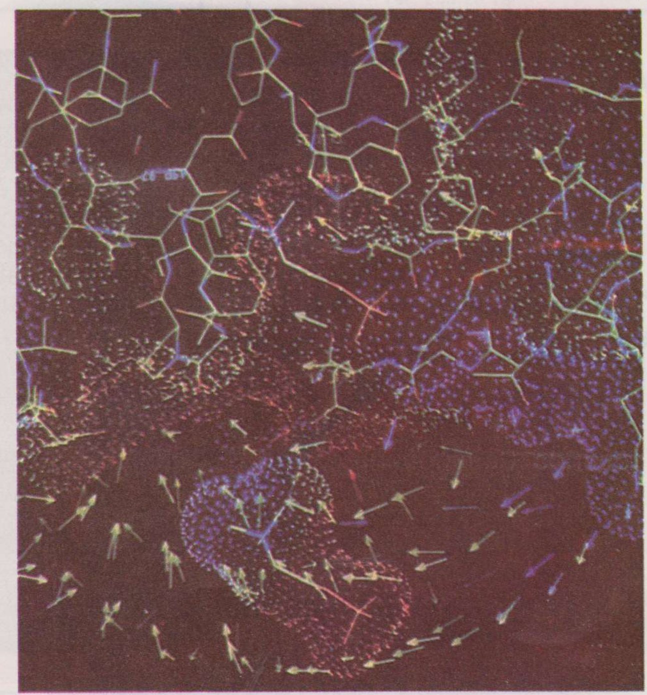
      (DERI Y x)
          (EQ Z  (COS Y * x)))
```

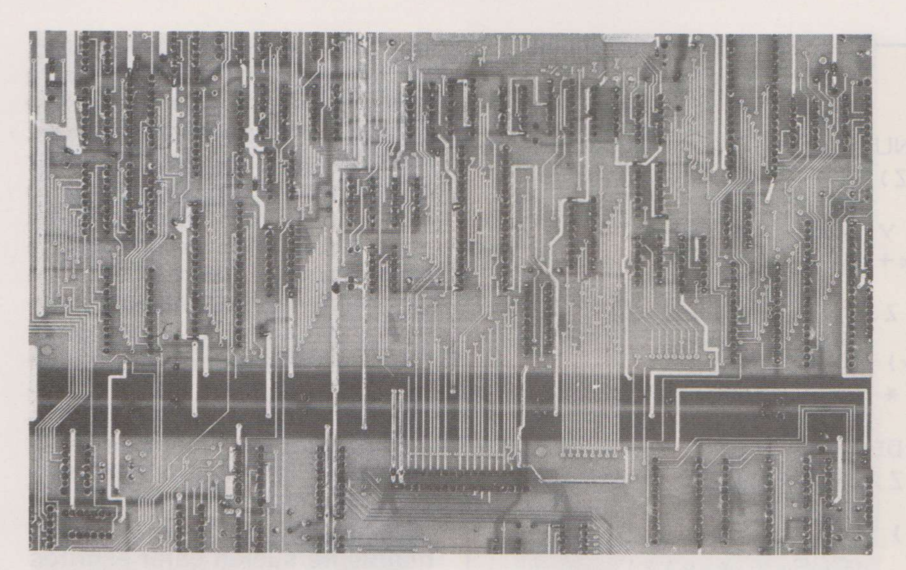
mente me arrincone, poniendo, primero, la torre, con lo que yo correría el rey, y después, la dama. Por ese camino parece que no tengo mucho futuro». —«Ah... pero si muevo al otro lado el rey, él tiene que mover el caballo, con lo que yo podría sacar mi dama y darle jaque. ¡Prepárate, tío!».

Es decir, hemos evaluado nuestras posibles jugadas, y después hemos pensado cómo jugaría él, y cómo jugaríamos nosotros después de su hipotética jugada. Evaluamos todas (muchas o algunas) de nuestras jugadas, y todas las posibles jugadas que el contrario puede hacer para cada una de las nuestras, y así sucesivamente, y al final nos quedamos con la que parece mejor. En el caso de arriba, sería la última de ellas. (Desgraciadamente suele pasar que B mueve un peón, con lo que el mate es inminente).

En los buenos programas de ordenador, el secreto está en cuál de todas las posibles jugadas evaluadas es la mejor y para eso se crean unas funciones que asignan una puntuación a cada movimiento, se llega a un nivel (mi jugada es el primer nivel, la respuesta del contrario el segundo nivel la siguiente mía el tercero...), normalmente suelen estar escritos en lenguajes que facilitan todo esto. Estos son lenguajes que manejan «listas», y de los cuales el primero fue el LISP, allá por el año 1960, y fue desarrollado por John McCharty; aquél que alguna vez haya trabajado con LOGO, habrá ya tenido relación con el proceso listas. Pero de todos ellos el más curioso es el PROLOG. Fue desarrollado en Francia, en Marsella, por Alain Colmerauer en 1972. Es por tanto muy reciente,







y su particularidad radica en que no tenemos que dar un montón de instrucciones que le dicen que A tiene que valer 22, después que se vaya incrementando B o que calcule o imprima C. Las instrucciones que se dan en programa PROLOG son simplemente «hechos que pasan», relaciones o, más propiamente, conocimientos. Nosotros somos algo así, una especie de paella de recuerdos, en la que cada grano es un dato (eso de ahí vuela), o una regla (los elefantes no vuelan, y, a partir de ahí, deducimos cosas (eso de ahí, por tanto, no es un elefante), y de forma similar actúa el PROLOG, metemos datos y reglas y después le preguntamos cosas. Este lenguaje, debido a su aplicación en inteligencia y procesamiento del lenguaje, fue el que escogieron los japoneses en un proyecto que va a durar unos 10 años, y en los que en ese tiempo quieren desarrollar una nueva máquina, casi un supercerebro. Es lo que ellos mismos llaman «la quinta generación» de ordenadores.

Para que veáis un ejemplo de cómo trabaja el PROLOG, en el cuadro 2 tenéis un ejemplo de un programa en este lenguaje. Su función es encontrar la derivada de una función cualquiera, y para eso se le han definido unas relaciones que aquel que haya estudiado algo sobre derivadas podrá sintetizarlas como sigue:

— la derivada de una constante es cero. Esa es la primera instrucción.

— La segunda es que la derivada de x es 1.

— La tercera nos dice cuál es la derivada de la suma.

Así se van definiendo esas reglas, que son las mismas que nosotros mismos tenemos.

A. de Mora-Losana

MASTERMIND

El juego tiene su base en los clásicos «Mastermind». En este caso el ordenador piensa un número y tú otro. El número ha de ser de tres cifras entre el 0 y el 9 inclusive, y no podrá repetirse la misma cifra.

Al principio del juego, el ordenador te pregunta quién empieza a jugar. Si comienza el ordenador a jugar te dirá el número que él cree que tú tienes y te pedirá el número de muertos (colocados en su lugar) y el número de heridos (descolocados); después te preguntará por el número que tú crees que ha pensado él y así sucesivamente. Es muy importante que sepas que en la primera jugada el ordenador PIENSA DURANTE TRES MINUTOS, en las restantes jugadas ya responde en segundos.

```
5 DIM N$(253,3)
10 GOSUB 900
20 GOSUB 600:FOR I=1 TO 3:D$(I)=A$(I):NEXT I
40 IF C$="S" THEN GOSUB 500
50 GOSUB 400
60 IF C$<>"S" THEN GOSUB 500:C$="S":GO TO 50
70 IF C$="S" THEN GOSUB 800
80 GOTO 50
299 REM ***** DA MUERTOS Y HERIDOS ****
300 MU=0:HE=0
```

GUIA PRACTICA

COMMODORE 64

SUPER QUINIELAS Disco
- Gestión total de los datos deportivos de 1.ª y 2.ª división.
- Creación de una quiniela estadística múltiple.
- Reducción al 13, al 12 y por figuras de una quiniela propia o estadística con el 100 % de acierto.
- Con salida por impresora y escrutinio .... **6.000** Ptas.

FAST TURBO MENUE
- Acelera el Datassette .... **8.500** Ptas.

FAST TURBO DISK
- Acelera la Unidad de Discos ..... **8.500** Ptas.

ASTOC-DATA, S. A. República Argentina, 40
Apt. 695 Santiago de Compostela Tel. (981) 59 95 33

DEFOREST
MICROINFORMATICA

TODO SOBRE **COMMODORE - 64** Y **VIC - 20**

LOS ULTIMOS JUEGOS EN EL MERCADO
TODO EN PERIFERICOS - LIBROS
PROGRAMAS DE GESTION - ETC.

SOLICITE INFORMACION POR CORREO

BARCELONA-15
C/ Viladomat, 105. Tel. 223 72 29

AMIGOS DEL COMMODORE 64K

COMPRA-VENTA
PROGRAMAS DE OCASION

Entre otros: COLOSSUS CHESS, PITS-TOP II, RAID OVER MOSCOW, MATCH POINT, DECATHLON, MANIC-MINER, ONE ON ONE, POLE POSITION, GOSTBUSTERS, SKYFOX, H.E.R.O. y 300 títulos más, pídenos el tuyo.

Por sólo **1.100** pts. más gastos de envío puedes conseguir tu programa de ocasión favorito, garantizados y comprobados.

Pídenos gratis nuestro catálogo de Programas.

¡Ah! y Programas con
PREMIO-OBSEQUIO

Rellena este cupón:
Deseo recibir contra reembolso:
Nombre del Programa ....

ME LO ENVIAN A:
D. ....
Calle .... N.º ....
Población ....
Código Postal .... Tel. ....

ENVIAR A: **AMIGOS DEL COMMODORE 64-K**
Apartado de Correos 34.155 BARCELONA

INTERFACE COPIADOR COSMOS THOUG (CBM 64 VIC 20 y C 128)

Este interface le permite hacer copias de seguridad de TODO software en soporte cassette (de CUALQUIER ordenador personal).

Funciona con CBM 64, VIC 20, C128 y dos CN2 o un CN2 y un cassette normal.

100 % de éxito GARANTIZADO en la copia. Testigo acústico (volumen regulable).
1 AÑO de garantía.
De exclusivo uso personal.
P.V.P. 4.600 ptas. incluido el IVA



DELTABIT
Colón, 20
SILLA (VALENCIA)
Tel. (96) 120 29 25

DISTRIBUIDORES BIENVENIDOS

COMPARE LOS PRECIOS

Commodore 16 23.900 Pts.
Joystick Quick Shot II 2.450 Pts.
Discos RPS 1D 340 Pts. Caja de plástico
Discos RPS 2D 395 Pts. Caja de plástico
Monitor Giaegi con sonido 22.900 Pts.

OFERTA COMMODORE 128
Consulte precios

ASTOC DATA, S.A.
C/ República Argentina, 40 Apto. 695
Santiago de Compostela
Tel. (981) 59 95 33

RADIO WATT

- COMPONENTES ELECTRONICOS
- COMMODORE Y AMSTRAD
- AMPLIA BIBLIOTECA TECNICA
- RADIO-COMUNICACIONES CB Y EQUIPOS HF/VHF/UHF

Paseo de Gracia, 126-130
Tel.: 237 11 82 - 08008 BARCELONA*



GOTO-55

Distribuidor Oficial de:



commodore
HARDWARE - SOFTWARE
LIBRERIA - CLUB DE SOFTWARE
ORDENADORES DE GESTION

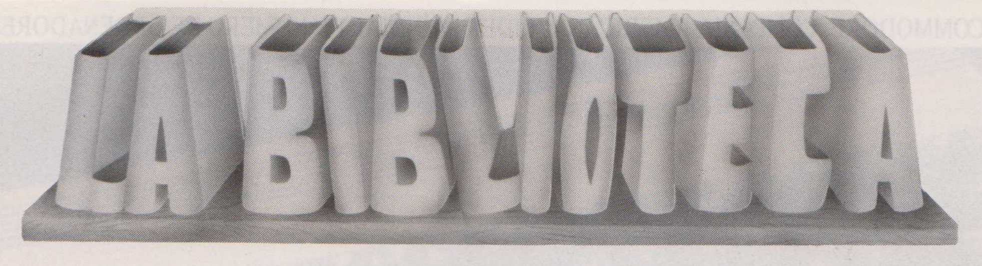
C/ Muntaner, 55 - 08011 BARCELONA
Tel.: 253 26 18

ANUNCIESE por MODULOS

prolog
para
- commodore
- amstrad
- spectrum

escribir a:
M&M. c/Vinateros, 12
28030-MADRID

```
310 FOR I=1 TO 3:IF A$(I)=B$(I) THEN MU=MU+1
320 NEXT I
330 FOR I=1 TO 3 :FOR J=1 TO 3
340 IF A$(I)=B$(J) THEN HE=HE+1:GOTO 360
350 NEXT J
360 NEXT I
370 HE=HE-MU
380 RETURN
399 REM ****** PIDE NUMEROS ***********
400 INPUT"NUMERO :";Z$:FORZ=1TO3:A$(Z)=MID$(Z$,Z,1):NEXT:GOSUB1000
410 FOR I=1 TO 3:B$(I)=D$(I):NEXT I
420 GOSUB 300
440 PRINT A$(1);A$(2);A$(3);" ";MU;" M ";HE;" H"
445 IF MU=3 THEN PRINT"▪    G A N A S T E !":END
450 RETURN
499 REM ******* JUEGA PRIMERO **********
500 GOSUB 600:GOSUB 700:MAXI=1
501 PRINT "ESTOY PENSANDO..."
505 FOR N=48 TO 57:A$(1)=CHR$(N)
510 FOR K=48 TO 57:A$(2)=CHR$(K)
520 FOR L=48 TO 57
530 IF N=K OR N=L OR L=K THEN GO TO 560
540 A$(3)=CHR$(L)
551 GOSUB 300
555 IF MU=MER AND HE=HRI THEN  FOR R=1 TO 3:N$(MAXI,R)=A$(R):NEXT R:MAXI=MAXI+1
560 NEXT L:NEXT K:NEXT N
565 MAXI=MAXI-1
570 GOSUB1000
580 RETURN
599 REM ******* GENERA NUMERO **********
600 FOR I=1 TO 3:A$(I)=MID$(STR$(INT(RND(1)*9)),2,1):NEXT I
620 IF A$(1)=A$(2) OR A$(2)=A$(3) OR A$(3)=A$(1) THEN GO TO 600
630 RETURN
699 REM *** PIDE MUERTOS Y HERIDOS ****
700 PRINT TAB(16)"NUMERO ";A$(1);A$(2);A$(3);"  MUERTOS";
701 INPUT MER
703 GOSUB 1000
710 PRINT TAB(16)"NUMERO ";A$(1);A$(2);A$(3);"  HERIDOS";
711 INPUT HRI
713 GOSUB 1000
725 IF MER=3 THEN PRINT"▪   I'M SORRY,PERO HE GANADO...":END
730 FOR R=1 TO 3:B$(R)=A$(R):NEXT R
740 RETURN
799 REM ******* ESCOGE NUMERO AZAR *****
800 IN=INT (MAXI*RND(1)+1):FORR=1TO3:A$(R)=N$(IN,R):NEXTR:GOSUB 700
805 T=1
810 FOR O=1 TO MAXI
820 FOR R=1 TO 3:A$(R)=N$(O,R):NEXT R
830 GOSUB 300
840 IF MU=MER AND HE=HRI THEN FOR R=1 TO 3:N$(T,R)=N$(O,R):NEXT R:T=T+1
860 NEXT O
870 MAXI=T-1
880 RETURN
899 REM ******* PRESENTACION ***********
900 PRINT "]▪            M A S T E R M I N D"
910 PRINT "          *******************":PRINT
911 PRINT"PIENSA UN NUMERO DE TRES CIFRAS          Y YO PENSARE OTRO...":PRINT
920 INPUT "EMPIEZO YO? (S/N) ";C$:GOSUB 1000
990 RETURN
1000 PRINT "]                                      ]":RETURN
```



C-64 TELECOMUNICACIONES
Autor: Jonathan Erickson.
Editorial: McGraw-Hill.
Traducción del inglés: 200 páginas.

La mayoría de nuestros lectores han utilizado el ordenador para jugar, aprender o programar, procesar textos, calcular, etc., pero sin duda son todavía minoría los que han explorado la comunicación de datos. Sin embargo, cuando a un ordenador se le añade un modem y un programa de comunicaciones, se abren para el usuario las puertas de un mundo insospechado, en el que la información es el elemento vital.

El contenido de este libro puede dividirse en dos partes claramente diferentes aunque no reflejadas en la estructura del libro. La primera parte recogería todo lo referente a la comunicación entre ordenadores en general y con el C-64 en particular, aplicaciones, bases de datos, modems, protocolos, software de comunicaciones etc., es decir, los conocimientos mínimos que debe tener un usuario antes de conectar su C-64 al teléfono.

La segunda parte está formada por una voluminosa información sobre los servicios a los que cualquier usuario de un C-64 norteamericano puede acceder desde su casa a través del teléfono. A cada tipo de servicio se dedica un capítulo:

— La compra desde casa.
— Servicios bancarios.
— Correo electrónico.
— Cursos de Universidad en casa.
— Bibliotecas informatizadas.
— Agencias de viajes, tarifas y reservas.
— Información bursátil.
— Sistemas de tablón de anuncios.
— Obtención gratis de programas.
— Telejuegos

y una larga serie de facilidades que han debido ya despertar la envidia de quien lee este comentario. En nuestro país esta cuestión lleva cierto retraso y la política seguida es mucho más restrictiva. Sin embargo,



conviene no despistarse y estar preparados, porque este tema va a sufrir un fuerte cambio en los próximos años. Mientras tanto, nada impide que los usuarios de «home-computers» españoles utilicen la línea telefónica con el correspondiente modem homologado y el software adecuado para comunicarse con otros usuarios incluso de máquinas distintas. Para los que tengan este propósito, el libro será una guía interesante.

ADVANCED PROGRAMMING TECHNIQUES ON C-64
Autor: David Lawrence.
Editorial: Sunshine Books.
V. original: 174 páginas.

David Lawrence es uno de los escritores sobre microordenadores más conocidos en Inglaterra, también es autor de varios programas comerciales y colaborador habitual en revistas. Este libro es uno de los mayores éxitos en su amplia lista de títulos publicados. En sus páginas analiza algunas de las técnicas requeridas en la programación comercial y por él mismo empleadas.

Si habéis intentado realizar un programa complejo, os habréis encontrado con numerosas dificultades para conseguir que funcione y estaréis sin duda convencidos de que programar es algo más que conocer los comandos y sentencias de un lenguaje. Pero si habéis ido un poco más lejos, y habéis compartido el desarrollo de programas con otros amigos o intentáis comercializarlos, sabréis que a un programa, además de funcionar, se le exigen otras cosas. Y ocurre, normalmente, que el que un programa cumpla o no estas otras condiciones, está en relación directa con las posibilidades de que el programa funcione correctamente.

En resumen, un buen programa debería ser modular, estructurado, claro, utilizar las alternativas más sencillas cuando existen varias, contener facilidades de depuración y mantenimiento,

y estar bien documentado. Además, desde el punto de vista del usuario, debe ser seguro, rápido, eficiente, fácil y agradable de manejar e ir acompañado de un buen manual de uso.

Para conseguir todo esto existen numerosas técnicas y una teoría de la programación en constante desarrollo.

En este libro se recopilan algunas de las técnicas empleadas por el autor en sus programas y muchos consejos sobre cómo hacer mejores programas en menos tiempo y con mayores garantías de éxito.

En concreto, los temas más extensamente tratados son los de modularidad, depuración y verificación, entrada y salida formateada



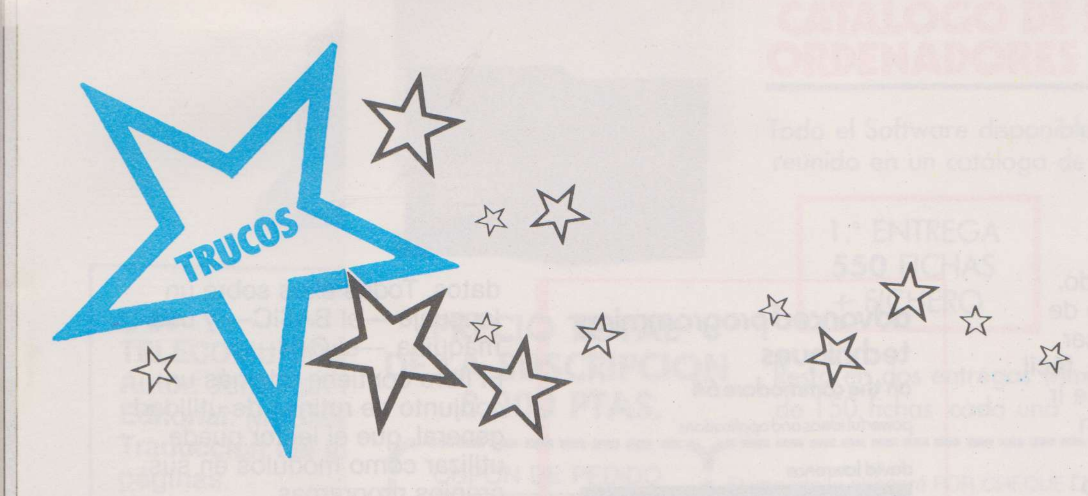
de información, tratamiento de cadenas de caracteres, gestión de errores, ordenación de valores y estructuras de datos. Todos ellos sobre un lenguaje —el BASIC— y una máquina —el C-64—.

El libro contiene además un conjunto de rutinas de utilidad general, que el lector puede utilizar como módulos en sus propios programas.

En el índice del libro existen dos ausencias importantes, justificadas por el autor en función de la extensión y complejidad de estos temas, que son las técnicas gráficas y las de procesamiento matemático.

Un libro que aporta muchas e interesantes ideas para quienes desean aumentar sus conocimientos de programación y mejorar la calidad de sus programas.

José M.ª Lizaso Azcárate



Ventanas en el C-16

Este pequeño programa está pensado para que sea posible trabajar con ventanas tanto en el C-16 como en el PLUS 4. Además, y al contrario que con muchas otras rutinas de creación de ventanas, con la que aquí os ofrecemos es posible salir de la ventana una vez creada o modificar las dimensiones de la misma desde el interior de un programa en ejecución.

La rutina, que incluye unas líneas de demostración, comienza llenando la pantalla de espacios en inverso, en color y en distintas tonalidades. A continuación se abre la primera ventana, se cambia de color y se repite la rutina de impresión. En este segundo caso, como la longitud de la línea dentro de la ventana es menor, se produce una especie de *scrolling* de los caracteres, lo que da lugar a unos interesantes efectos de movimiento.

```
5 COLORO,4,3
10 Z=2:C=0:X=5:Y=5:X1=34:Y1=20
20 FORI=1TO1000
30 COLOR1,Z,C
40 PRINT"■ ";
50 C=C+1:IFC>7THENC=0
60 NEXT
70 Z=Z+1
80 IFZ=3THENGOSUB63000
85 IFZ>3THENY=Y+1:X=X+1:X1=X1-1:PRINT"■■":GOSUB63000
89 PRINT"■"
90 GOTO 20
63000 :
63010 :
63020 CL$="■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
63030 CD$="■■■■■■■■■■■■■■■■■■■■■■■■■"
63040 PRINT"■";LEFT$(CD$,Y);LEFT$(CL$,X);CHR$(27);"T";
63050 PRINTLEFT$(CD$,(Y1-Y));LEFT$(CL$,(X1-X));CHR$(27);"B"
63060 RETURN
```

RESET

Aquí te mostramos unas llamadas SYS que producen un reset en tu ordenador, evitando apagar y encender éste para poder suprimir programas de la memoria sin suprimir las rutinas existentes en código máquina.

Ordenador	Reset
C-128	SYS 65341
C-64	SYS 64738
PLUS/4	SYS 65526
C-16	SYS 65526
VIC-20	SYS 64802

Cambio de colores con las teclas de función

Hemos recibido esta pequeña rutina, remitida por Carlos Hernández desde Barcelona, para nuestra sección de trucos, que insertada al principio de cualquier programa nos permitirá cambiar en cualquier momento el color del borde y fondo de la pantalla utilizando las teclas de función (F1-borde, F3-fondo).

La rutina quedará desactivada con las teclas RUN/STOP—RESTORE, y volverá a activarse al ejecutar el programa nuevamente.

```
10 FOR A=49152 TO 49183:READ B:POKE A,B:NEXT A
15 POKE 789,192:POKE 788,0
20 DATA 166,197,228,251,240,20,224,4,240,7,224,5,240,9,76,26,192
30 DATA 238,32,208,76,26,192,238,33,208,134,251,76,49,234,0
```



cartas

Problemas de grabación

Al querer grabar programas con el nuevo *cassette* que Commodore ha lanzado (el más moderno) tengo problemas, ya que mi ordenador produce un LOAD ERROR al poner cintas grabadas con el antiguo C2N 1530. ¿A qué puede ser debido? ¿Puede ser por tener ambos *cassettes* cabezales distintamente regulados? ¿Cómo regular los cabezales?

José Antonio Fillat
Lérida

Es muy frecuente este tipo de error, cuando nosotros salvamos programas con un *datassete* que tiene el cabezal desviado realizamos así una grabación defectuosa, que sólo puede ser leída por ese mismo *cassete*. De todas formas siempre es posible, regulando el cabezal, conseguir tener los dos *datassettes* con la misma inclinación en el cabezal. Actualmente se comercializan varios programas para regular los cabezales, entre ellos: Cinta Azimuth para alineación de la cabeza.

Memorias de Pantalla y Color

P: Tengo un Commodore 16 y me gustaría saber cómo puedo dar colores a los símbolos que se sitúan en pantalla cuando los coloco con POKEs, ya que hasta ahora lo he intentado, pero no entiendo cómo funcionan las Memorias de Pantalla y de Color, de las que habla el manual.

Anabel Barcenilla
Madrid

R: Tu pregunta no está muy clara, en relación con lo que quieres saber. Pero intentaremos aclarar todo lo referente a la Memoria de Pantalla y a la Memoria de Color que poseen los ordenadores Commodore.

Para situar en pantalla un símbolo o carácter, lo único que tienes que hacer es un POKE con la dirección del punto de pantalla donde lo quieres situar y luego, después de la coma, el valor del código ASCII correspondiente a ese símbolo o carácter.

La pantalla tiene 1.000 puntos en modo normal, es decir, 40 columnas por 25 líneas de pantalla.

Si luego quieres dar color al símbolo que has colocado en pantalla tienes que buscar el punto equivalente en el mapa de memoria de color, para escribir un POKE a dicha dirección, y después de la coma, el valor del color que quieres activar.

Las direcciones de memoria de Pantalla y de Color para los distintos ordenadores son:

Ordenador	Pantalla	Color
VIC	7680	38400
C-16	3072	2048
PLUS/4	3072	2048
C-64	1024	55296
C-128	1024	55296

Dos diskettes en uno

Quisiera preguntaros una cosa acerca de la unidad de disco, y los discos a utilizar. Me han dicho y he comprobado que a los discos de una cara y doble densidad se les puede practicar un pequeño corte y se convierten en discos de doble cara. ¿Esto es bueno para el disco o, en caso contrario, para la unidad de disco?

Una cosa que hecho a faltar en la revista son programas de interés o aplicación; no tanto juego. Podríais hacer un concurso permanente a la mejor aplicación y otro al mejor juego.

Miguel Ramón
Barcelona

La unidad de disco nunca podrá salir perjudicada con esta pequeña transformación, ya que los discos por su configuración física pueden recibir perfectamente información por ambas caras y el disco nunca perderá la información, siempre que efectúes el corte en el lugar oportuno. Conocemos a bastantes personas que utilizan este sistema y todavía no han tenido problemas.

Respecto a otra de las cuestiones que nos planteas, diremos que las aplicaciones, generalmente, están destinadas a las necesidades particulares de cada persona, o trabajo que desempeñen. De hecho, hemos creído conveniente comenzar una serie de artículos que traten sobre las posibilidades y modos de construirse una aplicación utilizando ficheros con *cassette,* unidad de disco, impresora, etc.

De todas formas ya sabéis que todo aquello que nos enviéis será publicado si responde al interés de todos los usuarios.

TV y Sprites

P: ¡Hola! Mi primera pregunta es saber si se puede estropear un televisor normal por tenerlo siempre enchufado al ordenador. La segunda va de *Sprites*. En el n.° 22 publicasteis lo de los *Sprites* múltiples. En la muestra, la línea 230 deja el tamaño pequeño a los *Sprites*. He intentado agrandarlos sin conseguirlo. Además he cambiado los DATA de la locomotora para hacer el vagón y no va. Desconozco la causa. Para finalizar este repertorio de ignorancias, quisiera saber cómo se pone un *Sprite* detrás de otro en lugar de sustituirlo. Gracias.

P.D.: Tengo un C-64.

Carles Amaya
Barcelona

R: Aunque se habló mucho de este tema, con el Primer boom del Ordenador, el sufrimiento de los TV por culpa de los fuertes contrastes en las pantallas que emiten los ordenadores, es ciertamente relativo y la experiencia ha demostrado que esto no es

real y que no debemos asustarnos por su uso.

Respecto a tu segunda pregunta, en la línea 230 del programa deberás escribir POKE V+23,3: POKE V+29,3 siendo el 3 el valor que agranda al *Sprite* 0 del nivel 1 y al *Sprite* 1 del nivel 2 (2+1=3). Pero ten en cuenta que al hacerlos más grandes tendrás que variar la proporción que existe en los valores de X1 y X2 pasando de 24 a 48. Y por último, para colocar vagones tienes que reformar el bucle de lectura, activar y poner punteros, y dar valores a las nuevas coordenadas que moverán los vagones.

Sprites

Hola, me gustaría saber qué es eso de los *Sprites* multicolores y los monocolores; cómo se puede hacer: tener al mismo tiempo más de ocho *sprites* y cómo manejarlos por la pantalla, y si no se pueden tener más de ocho *sprites* cómo lo hacen en los juegos profesionales para hacer caracteres de alta resolución por toda la pantalla haciendo dibujos, etc.

¿Qué es eso de choque *sprites* y choque de fondo y *sprite*? Gracias por la respuesta.

David García
Barcelona

Los *Sprites* son caracteres gráficos programables en Alta Resolución que ocupan 24 × 21 pixels (puntos o *bits)* y que pueden ser monocolores (un color de fondo y otro de *sprite)* o multicolores (un color de fondo y 3 en el *sprite)*.

Para mantener más de ocho *sprites* en pantalla, tiene que conocer código máquina y aprovechando las interrupciones del ordenador enviar en menos de un segundo varias pantallas distintas que superpuestas contienen más de 8 *sprites* (hasta 64 *sprites)*.



Existe en el Commodore y el modo bipmap, con el que que puedes dibujar punto a punto y en alta resolución los dibujos que tú desees.

Sobre las colisiones entre *sprites*, el ordenador tiene una dirección de memoria que es capaz de detectar cuándo se juntan dos *sprites* en pantalla y así nosotros utilizarla como respuesta al golpe (destrucción, sonido, rebote, deformación, etc.). También posee otra dirección que detecta cuándo se junta un *sprite* y un dato o dibujo en pantalla. Como ves, las respuestas son cortas, pero claras. No hemos querido entrar más en detalle, porque necesitaríamos toda la revista para explicarte la utilización y uso de todas tus cuestiones. En los números 21, 22 y 23 de Commodore Magazine puedes encontrar más información sobre estos temas.



Outlaws



Nos encontramos en el Lejano Oeste Americano. En pleno desierto, el paraíso de los proscritos.

Los habitantes de los ranchos tuvieron que abandonar sus casas, ya nadie podía vivir tranquilo, puesto que los bandidos, para los que no existe ley, habían arrasado todo cuanto encontraron a su paso.

Reinaba el temor y la desolación; nadie se atrevía a enfrentarse a tan despiadada banda. Necesitaban ayuda y sólo un hombre podía dársela: el jinete solitario. No tenía nombre ni pasado, nadie sabía de dónde salió, pero era necesario; era el único capaz de liberar a los pacíficos habitantes del valle de los tiranos malhechores.

Los enemigos son muchos, y el jinete solitario deberá usar todos sus recursos para luchar contra ellos. El terreno es árido y con múltiples obstáculos que el jinete deberá salvar para perseguir a los proscritos.



Pero nuestro héroe cuenta con un fiel aliado, su rápido Caballo Blanco, que cuando esté trotando será capaz de saltar sobre los troncos caídos, siempre que su jinete tire de las riendas adecuadamente.

Estos hombres fuera de la ley con los que tendrá que luchar el jinete solitario, no tienen ningún escrúpulo en utilizar cualquier artimaña para derribarle. Le perseguirán a caballo, y se esconderán en los ranchos que han tomado, para poder disparar sin apenas ser vistos. Pero el jinete solitario lo sabe, y se protege bajando la cabeza para no ser alcanzado por las balas de los proscritos.

Nuestro jinete puede moverse a derecha e izquierda. También puede disparar en cualquier dirección, y volver la cabeza para apuntar al enemigo si éste le ataca por detrás.

Muchos personajes se cruzarán en el camino: forajidos, indios, gente de los ranchos..., también aparecerán objetos y edificios, como hoteles, salones y cabañas de indios, a medida que vayas avanzando en el juego y matando a los enemigos.

En general, se trata de un juego sencillo que con un poco de práctica no presenta grandes complicaciones. El movimiento de los personajes, con una buena calidad de animación, y los distintos elementos que irán apareciendo en tu pantalla te mantendrán en el Lejano Oeste.

Los efectos sonoros de los cascos del caballo, los disparos y caídas están bien conseguidos, así como la música con la que comienza el juego.

Con un poco de práctica, pronto conseguirás acabar con buena parte de los proscritos y liberar a los habitantes del valle, que no dudarán en agradecerte tu hazaña.

FICHA DEL JUEGO

NOMBRE:	**OUTLAWS**
DE:	**ULTIMATE (ABC. SOFT)**
ORDENADOR:	**COMMODORE-64**
CONTROL:	**JOYSTICK**

PUNTUACION	1	2	3	4	5	6	7	8	9	10
ORIGINALIDAD						■				
ADICCION						■				
GRAFICOS							■			
SONIDO						■				
GENERAL						■				

Airwolf

Airwolf trata básicamente de una misión de rescate por medio de un helicóptero, pero bastante más complicado de lo que puede parecer en tan simple descripción.

Hace falta un piloto muy especial para hacer volar un aparato como Airwolf. Es un modelo muy ligero y sofisticado de helicóptero de combate. Está capacitado para alcanzar velocidades supersónicas, y cuenta con un sistema de armamento construido con los últimos avances tecnológicos.

Era necesario un aparato así, para enfrentarnos a una misión tan difícil como la nuestra. Se trata de liberar a un grupo de científicos que han sido apresados por el General Zanarov, una de las más perversas mentes del terrorismo mundial.

El General ha ideado un sistema subterráneo de defensa de su base, que la convierte en la más impenetrable prisión que el hombre ha conocido.

Consiste en un sistema de innumerables cavernas, que deberás explorar y conocer con Airwolf, para poder llevar a cabo la misión que se te ha encomendado.

Encerrados en la profundidad de este tortuoso laberinto, que se encuentra en el subsuelo del Desierto de Arizona, hay, como hemos dicho, un grupo de brillantes científicos, que el General tiene prisioneros para que le ayuden en su intento de dominar el mundo.

Tú, como piloto de Airwolf, debes rescatar a estos hombres, y superar la gran estructura defensiva de esta prisión subterránea creada por una poderosa mente, que es en realidad con quien tendrás que luchar. Así pues, tu razonamiento ha de ser también rápido y práctico, ya que buena parte del éxito de tu misión dependerá de ello.

Hay muchas cavernas que se comunican entre sí, y a las que podrás acceder disparando sobre determinados dispositivos secretos que deberás encontrar. Dependiendo del camino que elijas, se abrirán unas u otras paredes, que te permitirán acceder a distintas partes de la prisión. Pero habrá que tener cuidado, puesto que nuestro enemigo se ha encargado de poner algunas trampas perfectamente camufladas, en las que puedes quedar atrapado, y fracasar en el rescate.

En tu viaje por el subsuelo verás a los prisioneros que te hacen señas para que los salves. No intentes hacerlo hasta no estar seguro de que puedes acceder al lugar donde se encuentran, pues cada choque en tu helicóptero va deteriorando sus sistemas defensivos, y si colisionas en repetidas ocasiones, Airwolf se destruirá definitivamente. Por ello, creo que al principio es interesante intentar conocer el terreno. Es decir, averiguar las conexiones y posibles salidas de cada caverna, para posteriormente dedicarse al rescate de los científicos.

Airwolf no es un juego complicado, pero requiere cierto entrenamiento, y sobre todo suavidad y precisión en el manejo del *joystick,* pues es muy fácil chocar con las paredes de la cueva. Es un juego muy entretenido, y si te puede la curiosidad de conocer el laberinto del General, puedes pasar mucho tiempo ante tu ordenador.



FICHA DEL JUEGO

NOMBRE: **AIRWOLF**
DE: **ELITE (ABC. SOFT)**
ORDENADOR: **COMMODORE-64,**
CONTROL: **JOYSTICK**

PUNTUACION	1	2	3	4	5	6	7	8	9	10
ORIGINALIDAD						■				
ADICCION								■		
GRAFICOS						■				
SONIDO							■			
GENERAL							■			



Las aventuras de Bond... Basildon Bond

Si nunca has estado en unos estudios de televisión tendrás ocasión de visitarlos junto a Basildon Bond, y vivir con él una divertida aventura.

Bajo estrictas y confidenciales órdenes de un misterioso «P», Basildon Bond tendrá que rescatar a un conocido comediante, Russ, que ha sido capturado por un rival del mundo del espectáculo.

Pero el rescate no es fácil. Bond deberá revelar una serie de códigos secretos. Para ello tendrá que correr a través de las distintas habitaciones y pasillos de los estudios de televisión, y recoger las pistas que encuentre a su paso. Hay muchas habitaciones y posibilidades de combinar estas pistas.

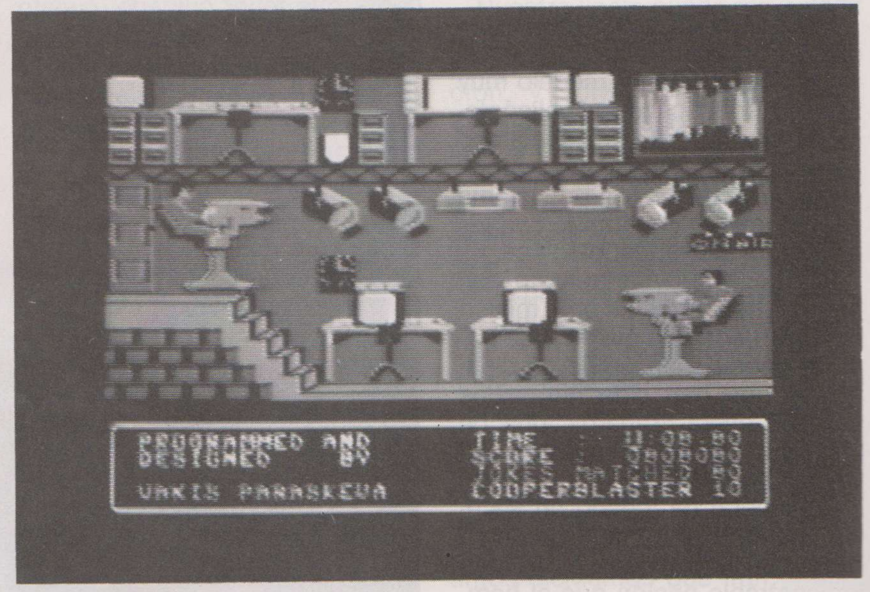
Bond tiene que llevar cada objeto que recoja al Cuarto de Computadoras, que es de donde parte su carrera, y si logra emparejar los objetos correctos, e introducirlos en el sistema de Ordenadores, éste le dará la clave para rescatar a Russ.

Al practicar, pronto te darás cuenta de que los objetos pueden ser utilizados para ejecutar diferentes funciones. Por ejemplo para volver al Cuarto de Computadoras, o para buscar en las diferentes habitaciones. Si mueves tu *joystick* hacia atrás, aparecerá un pequeño texto con una pista que puede serte de gran utilidad.

El tiempo máximo para el rescate es de cinco horas, y será suficiente si procuras evitar encuentros con las cámaras de televisión, que te harían regresar al principio del juego. Asimismo, pronto comprobarás que cierto incordiante ser te persigue constantemente, si practicas descubrirás sus intenciones.

Sin embargo, no estás tan solo en tu misión. Para librarte de las insistentes cámaras, que tratarán de hacerte perder tiempo, está Cooper Man, un extraño hombre volador que paralizará la acción cuando tú lo desees. Pero hasta con él tendrás que tener cuidado, pues también contiene a Blunder Woman, y ella puede causar más daño que beneficio.

Bond puede correr a derecha e izquierda, puede subir y bajar escaleras y coger y dejar objetos. También tiene habilidades para llamar a Cooper Man, o volver al Cuarto de Computadoras cuando lo crea conveniente. Con este repertorio de movimientos de nuestro amigo Basildon Bond, el único problema es controlar bien tu *joystick*, y agilizar tu memoria, para recordar las habitaciones ya visitadas y evitar perder tiempo volviendo a ellas. Si lo haces así, no te será difícil conseguir la clave para rescatar a Russ.



Mientras cargas el programa, podrás entretenerte observando la presentación del juego. En ella aparece un divertido primer plano de nuestro protagonista, Basildon Bond.

FICHA DEL JUEGO

NOMBRE: **THE ADVENTURES OF BOND**
DE: **PROBE SOFTWARE**
ORDENADOR: **COMMODORE-64, 128**
CONTROL: **JOYSTICK**

PUNTUACION	1	2	3	4	5	6	7	8	9	10
ORIGINALIDAD								X		
ADICCION						X				
GRAFICOS							X			
SONIDO						X				
GENERAL							X			



SPY VS SPY The Island Caper

Estamos ante un juego de estrategia y acción entre dos espías. «ESPIA CONTRA ESPIA» comienza presentándote las posibles opciones del juego, o bien jugar dos personas, o bien una y el ordenador. También tienes la posibilidad de elegir el nivel de dificultad del juego, cuando juegues contra el ordenador.

Dos espías aterrizan con sus paracaídas en una isla desierta presidida por un gran volcán que limitará el tiempo de la acción, debido a sus continuas erupciones. Ambos hombres tienen el mismo objetivo, pero sólo uno podrá conseguirlo. Su misión es montar las piezas de un proyectil, y conseguir llevárselo de la isla desierta, mediante un submarino que aparecerá en el momento preciso.

Nuestro personaje deberá ser hábil y rápido para superar a su contrincante. Sus jefes, que no ignoran lo peligroso de la misión, le proporcionaron una de las herramientas más revolucionarias en el campo del espionaje, el Manipulador de Trampas. Este moderno sistema dará periódicamente una clara información, por medio de unos mapas, de los lugares donde pueden encontrarse objetos útiles, o donde están las trampas que nos harán detenernos en el camino. Por supuesto, el espía contrincante tiene los mismos medios.

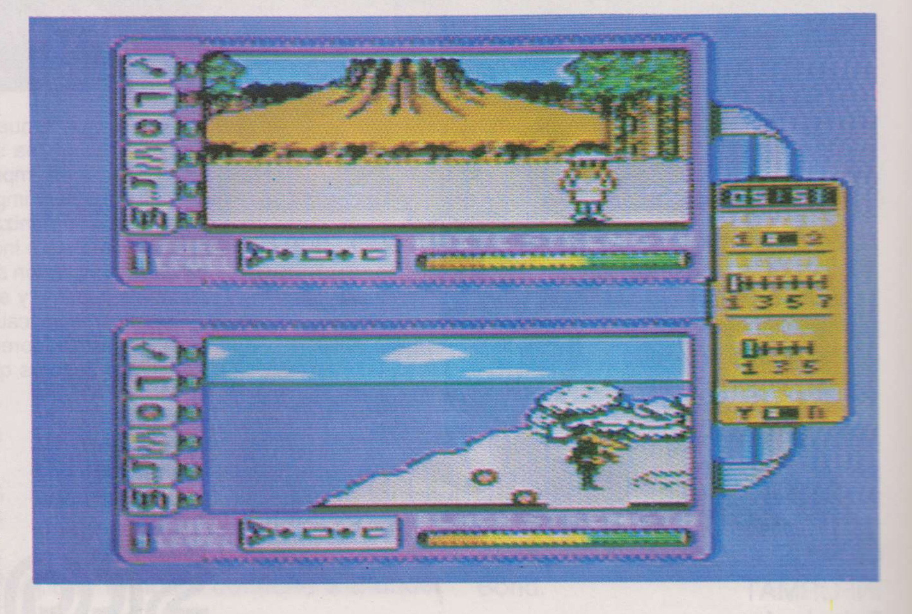
En la pantalla aparecerá una visión simultánea de la actividad de ambos espías. Pero cuidado, pues esto puede hacernos pensar que la isla es muy grande, y no es así. Pues no será difícil que te encuentres con tu oponente, y se produzca una cruenta lucha a muerte entre vosotros.

Durante tu misión a través de la isla encontrarás múltiples objetos y trampas. Debes tener cuidado con unas manchas oscuras que aparecen en el suelo, pues se trata de arenas movedizas, y el salir de ellas te costará una gran cantidad de energía. Si te adentras en el mar, en busca del submarino, procura estar atento a los tiburones, que si no tienes cuidado te engullirán de un bocado.

Pulsando dos veces el botón de disparo del *joystick* tendrás acceso completo al Manipulador de Trampas. Mediante él podrás ingeniártelas para hacer un poco más difícil la tarea a tu contrincante.

Hay muchos tipos de trampas: con lazo, fosos, pistolas, cocos-combate y otras. Podrás jugar muchas veces, y aprender a construir nuevas dificultades para tu enemigo.

Cuando consigas montar el proyectil, deberás ir hacia una de las costas, y allí aparecerá el submarino.



FICHA DEL JUEGO

NOMBRE:	**SPY VS SPY. THE ISLAND CAPER**
DE:	**FIST STAR SOFTWARE (ZAFIRO)**
ORDENADOR:	**COMMODORE-64**
CONTROL:	**JOYSTICK**

PUNTUACION	1	2	3	4	5	6	7	8	9	10
ORIGINALIDAD							■			
ADICCION						■				
GRAFICOS							■			
SONIDO								■		
GENERAL							■			

Moscas

VIC-20

Dentro del clásico tipo de juegos de «marcianos», Jaume M.ª Sagner, suscriptor de Commodore Magazine, nos remite desde Palamós una nueva versión llamada MOSCAS, por la semejanza de los «malos» con dichos insectos.

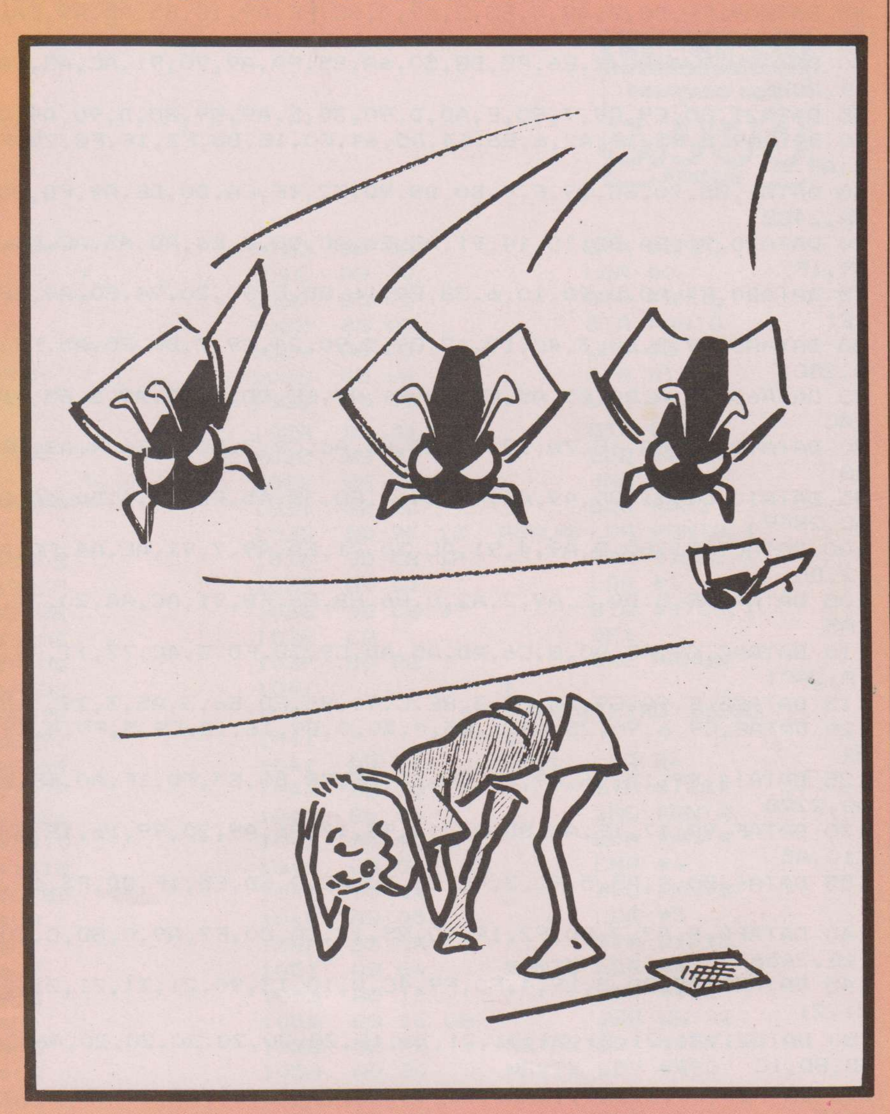
El juego consiste en una formación de moscas, con tres generales en su parte superior, que avanza bajando lentamente hacia el jugador, que es una nave situada en la parte inferior de la pantalla. Su misión es destruir a toda la formación y conseguir el mayor número de puntos.

Esto no es demasiado sencillo, pues las moscas salen espontáneamente de la formación, avalanzándose hacia la nave del jugador, disparando contra la misma y variando su dirección, lo que hace difícil su destrucción. Además, cuando la formación está casi totalmente destruida, los supervivientes se lanzan en masa contra la nave.

Si el jugador triunfa, a pesar de las dificultades, y logra eliminar la formación completa de moscas, aparece una nueva pantalla de mayor dificultad. A partir de la cuarta pantalla el juego se hace enormemente difícil, y lo único que interesa es sobrevivir.

El jugador dispone de tres naves y en todo momento en la parte inferior derecha de la pantalla aparece el tanteo que ha conseguido hasta el momento y en la izquierda el récord. Para controlar la nave se utilizan las teclas «ctrl» para disparos, y «:» y « = » para movimiento. Cuando finaliza el juego hay que pulsar la tecla «ctrl» para volver a intentarlo.

El programa BASIC se compone esencialmente de una serie de DATA'S donde se contiene en hexadecimal el programa en código máquina. Si se produce algún error al leer los DATA'S el programa avisa de ello. Al correr el programa hay que esperar a que se carguen los datos en las posiciones de memoria correspondientes, en lo que se tarda aproximadamente un minuto y medio. El programa así generado se puede grabar, de forma que cada vez que se desee jugar no hay que esperar a la generación de programa en máquina.



En el listado de lenguaje ensamblador que acompaña puede apreciar la función de cada rutina, siendo un interesante ejemplo de las posibilidades de programar en código máquina. □

```
10 DATA0,B,10,0,0,9E,35,31,32,31,0,0,0,A9,FF,8D,5,90,A9,8,8D,F,90,A9,F,8D,E
15 DATA90,85,A,A9,0,85,5,85,C,A9,3,85,4,A9,9,85,1,A9,93,20,D2,FF,A2,2C,BD,7F,13
20 DATA9D,CE,1F,A9,1,9D,CE,97,CA,D0,F2,A9,4,85,0,E6,C,A5,C,C9,4,90,4,A9,5,85,A,1
E4B
25 DATAA9,E4,A0,12,20,1E,CB,A2,FF,A0,20,88,D0,FD,CA,D0,F8,A5,5,F0,13,C6,5,F0,9,A
6,1
```

PROGRAMAS

```
30 DATAA9,1,9D,B8,1F,D0,4E,A9,20,A6,1,D0,1F,A5,CB,C9,2D,D0,9,A5,1,F0,2,C6,1,B8,5
0
35 DATAC,C9,2E,D0,8,A5,1,C9,14,F0,2,E6,1,A6,1,A9,7,9D,B8,1F,A9,1,9D,B8,97,A9,20,
264A
40 DATA9D,B7,1F,9D,B9,1F,AD,8D,2,29,4,D0,12,A5,2,D0,12,A9,0,9D,A2,1F,A9,7,9D,A2,
97
45 DATAA9,FF,D0,2,A9,0,85,2,A9,0,85,FE,A9,1D,85,AD,A9,EA,85,AC,A0,16,B1,AC,C9,2,
90
50 DATA9,C9,6,B0,2,E6,FE,B8,50,6A,85,FA,A9,20,91,AC,A5,FA,C9,1,F0,F1,A0,0,B1,AC,
C9,28BE
55 DATA21,B0,E9,C9,7,90,E,AD,D,90,30,5,A9,89,8D,D,90,A9,0,F0,39,C9,4,F0,40,84,FB
60 DATAA9,5,85,10,A2,6,86,64,A5,64,F0,1E,BD,F2,1F,F8,29,F,18,65,10,48,29,10,85,6
4,A9
65 DATA1,85,10,68,29,F,9,B0,D8,9D,F2,1F,CA,D0,DE,A9,FD,8D,D,90,A4,FB,A9,1,91,AC,
AA,24C8
70 DATA20,71,EA,BD,1D,14,91,AE,E6,AC,D0,2,E6,AD,A5,AC,C9,E4,F0,3,4C,D1,10,A5,AD,
C9,1F
75 DATAD0,F7,AD,D,90,10,6,38,E9,16,8D,D,90,20,94,E0,A9,1F,85,AD,A9,E4,85,AC,A0,0
,B1
80 DATAAC,C9,2,B0,3,4C,19,12,C9,5,90,24,C9,7,B0,F5,A5,FE,C9,4,90,13,A5,5,D0,EB,A
5.28D1
85 DATA63,25,A,D0,E5,A5,FD,D0,E1,A9,EE,8D,C,90,A9,2,85,FD,4C,E,12,85,FA,A9,20,91
,AC
90 DATA98,A6,FA,18,7D,18,14,A8,B1,AC,C9,7,D0,E,C6,4,A9,FF,8D,D,90,A9,20,85,5,4C,
58
95 DATA10,C9,21,B0,49,A5,FA,C9,4,F0,38,A5,FB,C9,4,D0,1A,84,FC,98,18,69,16,A8,B1,
AC,2849
100 DATAC9,21,B0,B,A9,4,91,AC,20,71,EA,A9,7,91,AE,A4,FC,A5,FA,A6,FB,E0,A,D0,E,C9
,2,D0
105 DATA4,A9,3,D0,2,A9,2,A2,0,86,FB,E6,FB,91,AC,AA,20,71,EA,BD,1D,14,91,AE,C6,AC
,A5
110 DATAAC,C9,FF,D0,8,C6,AD,A5,AD,C9,1D,F0,3,4C,77,11,A9,0,85,FD,A2,0,AD,C,90,10
,A,2A01
115 DATACE,C,90,C9,D5,B0,3,8E,C,90,86,FD,E6,3,A5,3,29,3,D0,50,A5,0,E6,0,C9,3,90
120 DATA8,C9,6,90,25,A9,1,85,0,A0,0,B9,16,1E,C9,5,F0,4,C9,6,D0,F,99,15,1E,AA,BD,
21
125 DATA14,99,15,96,A9,20,99,16,1E,C8,D0,E3,F0,1F,A0,FF,B9,16,1E,C9,5,F0,4,C9,6,
D0,229D
130 DATAF,99,17,1E,AA,BD,21,14,99,17,96,A9,20,99,16,1E,88,D0,E3,A5,FE,D0,3,4C,41
,10,A5
135 DATA4,D0,4,A5,5,F0,3,4C,58,10,A2,0,BD,E5,1F,DD,F3,1F,90,9,D0,12,E8,E0,6,D0,F
1
140 DATAF0,B,A2,7,BD,F2,1F,9D,95,13,CA,D0,F7,A9,0,8D,C,90,8D,D,90,A9,AB,A0,13,20
,1E,2456
145 DATACB,AD,8D,2,29,4,F0,F9,4C,D,10,13,90,21,21,21,21,21,21,21,21,21,21,21,21,
21,21
150 DATA21,21,21,21,21,21,21,21,1E,20,20,20,20,20,20,46,20,20,20,46,20,20,20,46,
8D,8D,1C
155 DATA20,20,20,20,45,20,45,20,45,20,45,20,45,20,45,20,45,8D,8D,1C,20,20,20,20,
107E
160 DATA45,20,45,20,45,20,45,20,45,20,45,20,45,8D,8D,1C,20,20,20,20,45,20,45,20,
45,20,45
165 DATA20,45,20,45,20,45,8D,8D,1C,20,20,45,20,45,20,45,20,45,20,45,20,45,20,45,
20,45,20
170 DATA45,8D,8D,1C,20,20,20,20,45,20,20,20,45,20,20,20,45,20,20,20,45,8D,90,8D,
8D,1200
175 DATA8D,8D,8D,8D,8D,8D,8D,8,0,21,92,85,83,8F,92,84,21,21,21,21,21,21,21,21,90
,95,8E,94
180 DATA8F,93,21,21,B0,B0,B0,B0,B0,B0,21,21,21,21,21,21,21,21,B0,B0,B0,B0,B0,B0,
21,13
185 DATA12,11,11,11,11,1D,1D,1D,1D,5,46,49,4E,1D,4A,55,45,47,4F,92,0,1947
190 DATA0,0,10,28,10,0,0,0,0,22,80,10,4,42,10,4,54,BA,FE,FE,D6,BA,92,44,2A,5D,7F
195 DATA7F,6B,5D,49,22,0,0,0,10,10,10,0,0,22,49,5D,6B,7F,7F,5D,2A,6B,77,6B,7F,3E
```

```
.1C,8
200 DATA8,8,1C,3E,1C,1C,1C,3E,6B,A2,0,8A,9D,0,1D,E8,E0,11,D0,F8,BD,B0,16ED
205 DATA13,9D,EF,1B,E8,E0,52,D0,F5,4C,D,10,15,17,16,7,1,3,3,1,3,3,3,2,2,7,667,00
210 FORJ=4096TO5160:READH$:D=0:FORI=1TOLEN(H$):X=ASC(MID$(H$,I))
215 D=D*16-(X-48)*(X>47ANDX<58)-(X-55)*(X>64ANDX<71):NEXT
220 IFD<256GOTO235
225 IFC-DTHENPRINT"ERROR ENTRE LINEAS"PEEK(63)-10"-"PEEK(63):E=E+1
230 C=0:J=J-1:GOTO240
235 POKEJ,D:C=C+D
240 NEXT:PRINTE"ERRORES"
250 IFETHENNEW
260 POKE45,40:POKE46,20:CLR
```



```
e33A                ;****************
033A                ;*              *
033A                ;*  MOSCAS V1   *
033A                ;*  ====== ==   *
033A                ;*              *
033A                ;* JAUME MARIA  *
033A                ;*    SAGUER    *
033A                ;*              *
033A                ;*  20-4-1985   *
033A                ;*              *
033A                ;****************

033A                RESOL EQU $9005
033A                COLOR EQU $900F
033A                VOLUM EQU $900E
033A                OSCIL EQU $900C
033A                RUIDO EQU $900D
033A                CROUT EQU $FFD2
033A                WSTRI EQU $CB1E
033A                ACOLO EQU $EA71
033A                RANDO EQU $E094
033A                PUNTO EQU $137F
033A                TACOL EQU $1421
033A                TABCO EQU $141D
033A                INCRE EQU $1418
033A                CTRL  EQU $028D
033A                DISCR EQU $00
033A                CONAV EQU $01
033A                TITIR EQU $02
033A                TISCR EQU $03
033A                VIDAS EQU $04
033A                TIMUE EQU $05
033A                DIFIC EQU $0A
033A                PANTA EQU $0C
033A                TIEMP EQU $FA
033A                TECLA EQU $CB
033A                ;
033A                ;
033A                ;
033A                ;*INICIALIZACION
033A                ;
033A                      ORG $100D
100D  A9 FF         INIT  LDA #$FF
100F  8D 05 90            STA RESOL
1012  A9 08               LDA #$08
1014  8D 0F 90            STA COLOR
1017  A9 0F               LDA #$0F
1019  8D 0E 90            STA VOLUM
101C  85 0A               STA DIFIC
101E  A9 00               LDA #0
1020  85 05               STA TIMUE
1022  85 0C               STA PANTA
1024  A9 03               LDA #3
1026  85 04               STA VIDAS
1028  A9 09               LDA #9
102A  85 01               STA CONAV
102C  A9 93               LDA #$93
102E  20 D2 FF            JSR CROUT
1031  A2 2C               LDX #$2C
1033  BD 7F 13      ARPUN LDA PUNTO,X
1036  9D CE 1F            STA $1FCE,X
1039  A9 01               LDA #1
103B  9D CE 97            STA $97CE,X
103E  CA                  DEX
103F  D0 F2               BNE ARPUN
1041                ;
1041                ;*PRINTCIPO JUEGO
1041                ;
1041  A9 04         PRI1  LDA #4
1043  85 00               STA DISCR
1045  E6 0C               INC PANTA
1047  A5 0C               LDA PANTA
1049  C9 04               CMP #4
104B  90 04               BCC PANTB
104D  A9 05               LDA #5
104F  85 0A               STA DIFIC
1051  A9 E4         PANTB LDA #$E4
1053  A0 12               LDY #$12
1055  20 1E CB            JSR WSTRI
1058  A2 FF         RETAR LDX #$FF
105A  A0 20         NEXTX LDY #$20
105C  88            NEXTY DEY
105D  D0 FD               BNE NEXTY
105F  CA                  DEX
1060  D0 F8               BNE NEXTX
1062  A5 05               LDA TIMUE
1064  F0 13               BEQ MITEC
1066  C6 05               DEC TIMUE
1068  F0 09               BEQ LOO
106A  A6 01               LDX CONAV
106C  A9 01               LDA #1
106E  9D B8 1F            STA $1FB8,X
1071  D0 4E               BNE LO1
1073  A9 20         LOO   LDA #$20
1075  A6 01               LDX CONAV
1077  D0 1F               BNE PRINT
```

PROGRAMAS

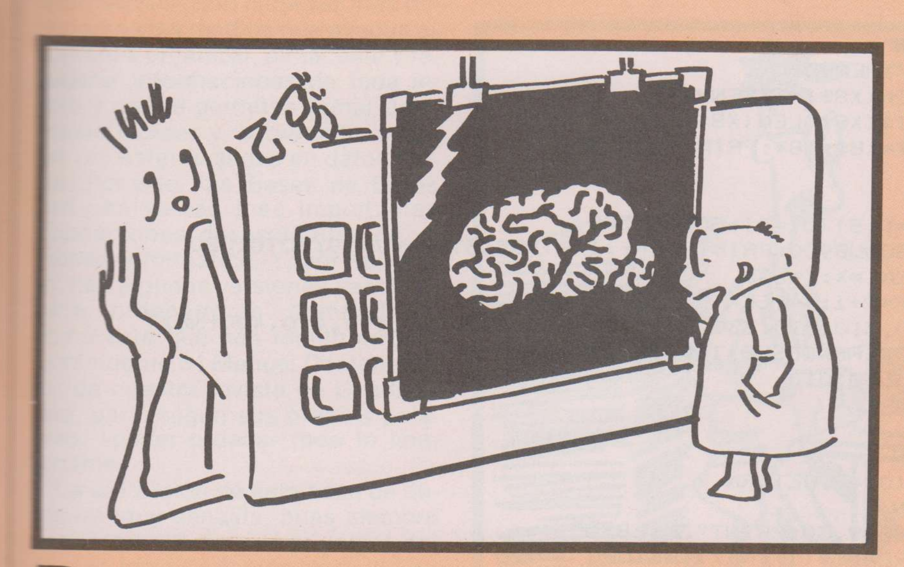
```
1079                  ;****************
1079                  ;
1079                  ;  MUEVE NAVE
1079                  ;
1079                  ; $2E=DERECHA
1079                  ; $2D=IZQUIERDA
1079                  ;
1079                  ;****************
1079                  ;
1079  A5 CB           MITEC LDA TECLA
107B  C9 2D                 CMP #$2D
107D  D0 09                 BNE DERE
107F  A5 01                 LDA CONAV
1081  F0 02                 BEQ NOPU
1083  C6 01                 DEC CONAV
1085  B8              NOPU  CLV
1086  50 0C                 BVC MUEVN
1088  C9 2E           DERE  CMP #$2E
108A  D0 08                 BNE MUEVN
108C  A5 01                 LDA CONAV
108E  C9 14                 CMP #$14
1090  F0 02                 BEQ MUEVN
1092  E6 01                 INC CONAV
1094  A6 01           MUEVN LDX CONAV
1096  A9 07                 LDA #7
1098  9D B8 1F        PRINT STA $1FB8,X
109B  A9 01                 LDA #1
109D  9D B8 97              STA $97B8,X
10A0  A9 20                 LDA #$20
10A2  9D B7 1F              STA $1FB7,X
10A5  9D B9 1F              STA $1FB9,X
10A8                  ;================
10A8                  ;
10A8                  ;    DISPARA
10A8                  ;
10A8                  ;
10A8                  ;===============
10A8  AD 8D 02              LDA CTRL
10AB  29 04                 AND #4
10AD  D0 12                 BNE L01
10AF  A5 02                 LDA TITIR
10B1  D0 12                 BNE L02
10B3  A9 00                 LDA #0
10B5  9D A2 1F              STA $1FA2,X
10B8  A9 07                 LDA #7
10BA  9D A2 97              STA $97A2,X
10BD  A9 FF                 LDA #$FF
10BF  D0 02                 BNE GUTI
10C1  A9 00           L01   LDA #0
10C3  85 02           GUTI  STA TITIR
10C5                  ;****************
10C5                  ;
10C5                  ;SUBE TIROS NAVE
10C5                  ;
10C5                  ;DA PUNTOS Y MATA
10C5                  ;
10C5                  ;MOSCAS.
10C5                  ;
10C5                  ;****************
10C5                  ;
10C5  A9 00           L02   LDA #0
10C7  85 FE                 STA $FE
10C9  A9 1D                 LDA #$1D
10CB  85 AD                 STA $AD
10CD  A9 EA                 LDA #$EA
10CF  85 AC                 STA $AC
```

```
10D1  A0 16           ABAJO LDY #$16
10D3  B1 AC                 LDA ($AC),Y
10D5  C9 02                 CMP #2
10D7  90 09                 BCC BORR1
10D9  C9 06                 CMP #6
10DB  B0 02                 BCS NBOR1
10DD  E6 FE                 INC $FE
10DF  B8              NBOR1 CLV
10E0  50 6A                 BVC NORM1
10E2  85 FA           BORR1 STA TIEMP
10E4  A9 20                 LDA #$20
10E6  91 AC                 STA ($AC),Y
10E8  A5 FA                 LDA TIEMP
10EA  C9 01                 CMP #1
10EC  F0 F1                 BEQ NBOR1
10EE  A0 00                 LDY #0
10F0  B1 AC                 LDA ($AC),Y
10F2  C9 21                 CMP #$21
10F4  B0 E9                 BCS NBOR1
10F6  C9 07                 CMP #7
10F8  90 0E                 BCC L03
10FA  AD 0D 90              LDA RUIDO
10FD  30 05                 BMI NRUI1
10FF  A9 89                 LDA #$89
1101  8D 0D 90              STA RUIDO
1104  A9 00           NRUI1 LDA #0
1106  F0 39                 BEQ ESCR1
1108  C9 04           L03   CMP #4
110A  F0 40                 BEQ NORM1
110C  84 FB                 STY $FB
110E  A9 05                 LDA #5
1110  85 10                 STA $10
1112  A2 06                 LDX #6
1114  86 64                 STX $64
1116  A5 64           WPUNT LDA $64
1118  F0 1E                 BEQ PUFF
111A  BD F2 1F              LDA $1FF2,X
111D  F8                    SED
111E  29 0F                 AND #$0F
1120  18                    CLC
1121  65 10                 ADC $10
1123  48                    PHA
1124  29 10                 AND #$10
1126  85 64                 STA $64
1128  A9 01                 LDA #1
112A  85 10                 STA $10
112C  68                    PLA
112D  29 0F                 AND #$0F
112F  09 B0                 ORA #$B0
1131  D8                    CLD
1132  9D F2 1F              STA $1FF2,X
1135  CA                    DEX
1136  D0 DE                 BNE WPUNT
1138  A9 FD           PUFF  LDA #$FD
113A  8D 0D 90              STA RUIDO
113D  A4 FB                 LDY $FB
11EF  A9 01                 LDA #1
1141  91 AC           ESCR1 STA ($AC),Y
1143  AA                    TAX
1144  20 71 EA              JSR ACOLO
1147  BD 1D 14              LDA TABCO,X
114A  91 AE                 STA ($AE),Y
114C  E6 AC           NORM1 INC $AC
114E  D0 02                 BNE NCER1
1150  E6 AD                 INC $AD
1152  A5 AC           NCER1 LDA $AC
```

```
1154  C9 E4              CMP #$E4
1156  F0 03              BEQ CERO1
1158  4C D1 10   JMABA   JMP ABAJO
115B  A5 AD      CERO1   LDA $AD
115D  C9 1F              CMP #$1F
115F  D0 F7              BNE JMABA
1161  AD 0D 90           LDA RUIDO
1164  10 06              BPL RUOFF
1166  38                 SEC
1167  E9 16              SBC #$16
1169  8D 0D 90           STA RUIDO
116C             ;****************
116C             ;
116C             ;BAJA MOSCAS TIRA
116C             ;
116C             ;TIROS-MOSCAS
116C             ;
116C             ;****************
116C             ;
116C  20 94 E0   RUOFF   JSR RANDO
116F  A9 1F              LDA #$1F
1171  85 AD              STA $AD
1173  A9 E4              LDA #$E4
1175  85 AC              STA $AC
1177  A0 00      SUBE    LDY #0
1179  B1 AC              LDA ($AC),Y
117B  C9 02              CMP #2
117D  B0 03              BCS HAZ
117F  4C 19 12   JMNO    JMP NORM2
1182  C9 05      HAZ     CMP #5
1184  90 24              BCC BORR2
1186  C9 07              CMP #7
1188  B0 F5              BCS JMNO
118A  A5 FE              LDA $FE
118C  C9 04              CMP #4
118E  90 13              BCC L10
1190  A5 05              LDA TIMUE
1192  D0 EB              BNE JMNO
1194  A5 63              LDA $63
1196  25 0A              AND DIFIC
1198  D0 E5              BNE JMNO
119A  A5 FD              LDA $FD
119C  D0 E1              BNE JMNO
119E  A9 EE              LDA #$EE
11A0  8D 0C 90           STA OSCIL
11A3  A9 02      L10     LDA #2
11A5  85 FD              STA $FD
11A7  4C 0E 12           JMP ESCR2
11AA  85 FA      BORR2   STA TIEMP
11AC  A9 20              LDA #$20
11AE  91 AC              STA ($AC),Y
11B0  98                 TYA
11B1  A6 FA              LDX TIEMP
11B3  18                 CLC
11B4  7D 18 14           ADC INCRE,X
11B7  A8                 TAY
11B8  B1 AC              LDA ($AC),Y
11BA  C9 07              CMP #7
11BC  D0 0E              BNE NO7
11BE  C6 04              DEC VIDAS
11C0  A9 FF              LDA #$FF
11C2  8D 0D 90           STA RUIDO
11C5  A9 20              LDA #$20
11C7  85 05              STA TIMUE
11C9  4C 58 10           JMP RETAR
11CC  C9 21      NO7     CMP #$21
11CE  B0 49              BCS NORM2
11D0  A5 FA              LDA TIEMP
11D2  C9 04              CMP #4
11D4  F0 38              BEQ ESCR2
11D6  A5 FB              LDA $FB
11D8  C9 04              CMP #4
11DA  D0 1A              BNE NOTIR
11DC  84 FC              STY $FC
11DE  98                 TYA
11DF  18                 CLC
11E0  69 16              ADC #$16
11E2  A8                 TAY
11E3  B1 AC              LDA ($AC),Y
11E5  C9 21              CMP #$21
11E7  B0 0B              BCS NTIRO
11E9  A9 04              LDA #$04
11EB  91 AC              STA ($AC),Y
11ED  20 71 EA           JSR ACOLO
11F0  A9 07              LDA #7
11F2  91 AE              STA ($AE),Y
11F4  A4 FC      NTIRO   LDY $FC
11F6  A5 FA      NOTIR   LDA TIEMP
11F8  A6 FB              LDX $FB
11FA  E0 0A              CPX #$0A
11FC  D0 0E              BNE INC
11FE  C9 02              CMP #2
1200  D0 04              BNE IZQ
1202  A9 03              LDA #3
1204  D0 02              BNE ESCR3
1206  A9 02      IZQ     LDA #2
1208  A2 00      ESCR3   LDX #0
120A  86 FB              STX $FB
120C  E6 FB      INC     INC $FB
120E  91 AC      ESCR2   STA ($AC),Y
1210  AA                 TAX
1211  20 71 EA           JSR ACOLO
1214  BD 1D 14           LDA TABCO,X
1217  91 AE              STA ($AE),Y
1219  C6 AC      NORM2   DEC $AC
121B  A5 AC              LDA $AC
121D  C9 FF              CMP #$FF
121F  D0 08              BNE JM2
1221  C6 AD              DEC $AD
1223  A5 AD              LDA $AD
1225  C9 1D              CMP #$1D
1227  F0 03              BEQ CERO5
1229  4C 77 11   JM2     JMP SUBE
122C  A9 00      CERO5   LDA #0
122E  85 FD              STA $FD
1230  A2 00              LDX #0
1232  AD 0C 90           LDA OSCIL
1235  10 0A              BPL NOOSC
1237  CE 0C 90           DEC OSCIL
123A  C9 D5              CMP #$D5
123C  B0 03              BCS NOOSC
123E  8E 0C 90           STX OSCIL
1241  86 FD      NOOSC   STX $FD
1243             ;
1243             ;****SCROLLS*****
1243             ;
1243             ;DE LA FORMACION
1243             ;HACIA LA IZQUI.
1243             ;O A LA DERECHA.
1243             ;
1243             ;IGNORA LO QUE NO
1243             ;SON MOSCAS.
```

```
1243                  ;
1243                  ;****************
1243                  ;
1243  E6 03                 INC TISCR
1245  A5 03                 LDA TISCR
1247  29 03                 AND #3
1249  D0 50                 BNE L15
124B  A5 00                 LDA DISCR
124D  E6 00                 INC DISCR
124F  C9 03                 CMP #3
1251  90 08                 BCC SIZQU
1253  C9 06                 CMP #6
1255  90 25                 BCC SDERE
1257  A9 01                 LDA #1
1259  85 00                 STA DISCR
125B                  ;===============
125B                  ;
125B                  ;SCROLL IZQUIERDA
125B                  ;
125B                  ;===============
125B                  ;
125B                  ;
125B  A0 00           SIZQU LDY #0
125D  B9 16 1E        NXSCI LDA $1E16,Y
1260  C9 05                 CMP #5
1262  F0 04                 BEQ SCIZQ
1264  C9 06                 CMP #6
1266  D0 0F                 BNE NSCIZ
1268  99 15 1E        SCIZQ STA $1315,Y
126B  AA                    TAX
126C  BD 21 14              LDA TACOL,X
126F  99 15 96              STA $9615,Y
1272  A9 20                 LDA #$20
1274  99 16 1E              STA $1E16,Y
1277  C8              NSCIZ INY
1278  D0 E3                 BNE NXSCI
127A  F0 1F                 BEQ L15
127C                  ;
127C                  ;==============
127C                  ;
127C                  ;SCROLL DERECHA
127C                  ;
127C                  ;==============
127C                  ;
127C                  ;
127C  A0 FF           SDERE LDY #$FF
127E  B9 16 1E        NXSCD LDA $1E16,Y
1281  C9 05                 CMP #5
1283  F0 04                 BEQ SCDER
1285  C9 06                 CMP #6
1287  D0 0F                 BNE NSCDE
1289  99 17 1E        SCDER STA $1E17,Y
128C  AA                    TAX
128D  BD 21 14              LDA TACOL,X
1290  99 17 96              STA $9617,Y
1293  A9 20                 LDA #$20
1295  99 16 1E              STA $1E16,Y
1298  88              NSCDE DEY
1299  D0 E3                 BNE NXSCD
129B                  ;****************
129B                  ;
129B                  ;COMPROBACION DE
129B                  ;
129B                  ;FIN DE JUEGO.
129B                  ;
129B                  ;****************
129B                  ;
129B  A5 FE           L15   LDA $FE
129D  D0 03                 BNE NJM4
129F  4C 41 10              JMP PRI1
12A2  A5 04           NJM4  LDA VIDAS
12A4  D0 04                 BNE JMP5
12A6  A5 05                 LDA TIMUE
12A8  F0 03                 BEQ NJM6
12AA  4C 58 10        JMP5  JMP RETAR
12AD                  ;===============
12AD                  ;
12AD                  ;   FIN JUEGO.
12AD                  ;
12AD                  ;===============
12AD                  ;
12AD  A2 00           NJM6  LDX #0
12AF  BD E5 1F        NPUN  LDA $1FE5,X
12B2  DD F3 1F              CMP $1FF3,X
12B5  90 09                 BCC RECOR
12B7  D0 12                 BNE NRECO
12B9  E8                    INX
12BA  E0 06                 CPX #6
12BC  D0 F1                 BNE NPUN
12BE  F0 0B                 BEQ NRECO
12C0  A2 07           RECOR LDX #7
12C2  BD F2 1F        PASS  LDA $1FF2,X
12C5  9D 95 13              STA $1395,X
12C8  CA                    DEX
12C9  D0 F7                 BNE PASS
12CB  A9 00           NRECO LDA #0
12CD  8D 0C 90              STA OSCIL
12D0  8D 0D 90              STA RUIDO
12D3  A9 AB                 LDA #$AB
12D5  A0 13                 LDY #$13
12D7  20 1E CB              JSR WSTRI
12DA  AD 8D 02        GOVER LDA CTRL
12DD  29 04                 AND #4
12DF  F0 F9                 BEQ GOVER
12E1  4C 0D 10              JMP INIT
12E4                  ;
12E4                  ;***************
12E4                  ;
12E4                  ; PROGRAMA QUE
12E4                  ;
12E4                  ;     PREPARA
12E4                  ;
12E4                  ;  CARACTERES.
12E4                  ;
12E4                  ;***************
12E4                  ;
12E4                        ORG $1401
1401  A2 00                 LDX #0
1403  8A                    TXA
1404  9D 00 1D        NXBY1 STA $1D00,X
1407  E8                    INX
1408  E0 11                 CPX #$11
140A  D0 F8                 BNE NXBY1
140C  BD B0 13        NXBY2 LDA $13B0,X
140F  9D EF 1B              STA $1BEF,X
1412  E8                    INX
1413  E0 52                 CPX #$52
1415  D0 F5                 BNE NXBY2
1417  4C 0D 10              JMP INIT
141A                  ;* FIN PROGRAMA
```

Desensamblador para VIC-20



Desde Santiago de Chile, Eduardo Ahumada nos ha remitido este desensamblador para el VIC-20 estándar, que será de sumo interés para todos aquellos aficionados al lenguaje máquina.

Este programa, escrito en BASIC con un total de 5 K de RAM, junto con un mapa de memoria del ordenador, te permitirá acceder y examinar partes del Sistema Operativo, así como ver el funcionamiento de misteriosas subrutinas que tienen algunos programas.

Para todos aquellos que hayan oído hablar de desensambladores, pero nunca hayan trabajado con uno o incluso no sepan su utilización, diremos que la base de este tipo de programas es traducir código máquina a lenguaje ensamblador. Es decir, lee los contenidos de posiciones consecutivos de la memoria y «traduce» dichos contenidos a instrucciones en lenguaje ensamblador. Este lenguaje es el más próximo a código máquina, y en él, cada instrucción de las que puede ejecutar directamente el microordenador viene representada por un conjunto de letras denominadas nemónicos.

Una vez tecleado y ejecutado el programa, nos pide la dirección de memoria a partir de la cual se desea comenzar a desensamblar. A continuación se comienza a imprimir en la pantalla las instrucciones en ensamblador, hasta completar una página (17 instrucciones). En cada línea se imprime la dirección decimal de la instrucción, su código nemotécnico y sus operandos, si los hay.

Por supuesto, existe la posibilidad de que alguna posición de memoria no contenga ninguna instrucción válida, en cuyo caso el programa le asignará el código «CON» (constante) e imprimirá su valor decimal y el carácter ASCII correspondiente.

Una vez completada una página, y mediante las teclas de función, se podrá pasar a la página siguiente —F7— o retroceder a la página anterior —F1—. Incluso se podrá cortar este listado y empezar a desensamblar a partir de una nueva dirección pulsando F2, mediante la cual volveremos a la pantalla inicial, lugar donde se solicitará la posición de memoria.

Para lograr el efecto de retroceso, es necesario que el programa recuerde cuál era la dirección de la primera instrucción desensamblada de cada página. Esta dirección es recordada en la tabla S, que como tiene 10 posiciones, puede «recordar» hasta 10 páginas.




```
10 DEFFNA(X)=PEEK(X+1)*256+Y
100 GOSUB2000:GOTO1000
300 PRINT" #"MID$(STR$(Y),2,3);
301 P=P+1:RETURN
310 IFX<>255THENPRINTY;:GOTO301
312 J=I:IFI<35ORI>127THENI=46
313 PRINT" "CHR$(34);CHR$(I);CHR$(34)"     "J;:GOTO301
320 PRINTY"■,X";:GOTO301
330 PRINTY"■,Y";:GOTO301
340 PRINTFNA(P);
341 P=P+2:RETURN
```

PROGRAMAS

```
350 PRINTFNA(P)"▮,X";:GOTO341
360 PRINTFNA(P)"▮,Y";:GOTO341
370 PRINT" ("MID$(STR$(Y),2,3)",X)";:GOTO301
380 PRINT" ("MID$(STR$(Y),2,3)"),Y";:GOTO301
390 PRINT" ("MID$(STR$(FNA(P)),2,5)")";:GOTO341
400 X=Y:IFX>=128THENX=X-256
401 PRINTP+1+X;:GOTO301
410 RETURN
900 PRINT"▮"LEFT$("▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪",F7);TAB(C7);:RETURN
930 GETT8$:IFT8$=""THEN930
931 T8=ASC(T8$):RETURN
950 GOSUB900:X8$="":PRINT"_▮";
951 GOSUB930:IFT8=13ORT8=137THEN956
952 IF(T8<>157ANDT8<>20)ORLEN(X8$)=0THEN954
953 PRINT"▮  ▮▮_▮";:X8$=LEFT$(X8$,LEN(X8$)-1):GOTO951
954 IFT8>=48ANDT8<=57THENX8$=X8$+T8$:PRINTT8$"_▮";
955 IFLEN(X8$)<L7THEN951
956 PRINT" ";:RETURN
1000 GOSUB5100:C7=0:P=DI:PS=0:ST(0)=DI:GOSUB5200
1010 GOSUB5000:FORF7=2TO18:GOSUB900:PRINT"▮"P;:IFP>65535THENF7=19:GOTO1060
1015 X=PEEK(P):IFI$(X)=""THENI=X:X=255
1020 PRINTRIGHT$(I$(X),3);:P=P+1:Y=PEEK(P)
1030 ONASC(I$(X))-64GOSUB300,310,320,330,340,350,360,370,380,390,410,400:NEXT
1060 GOSUB930:IFT8=133ANDPS>0THENGOSUB1100:GOTO1010
1070 IFT8=136THENGOSUB1200:GOTO1010
1080 IFT8=137THEN1000
1090 GOTO1060
1100 PS=PS-1:P=ST(PS):RETURN
1200 IFPS=9THENDI=P:PS=0:ST(0)=P:RETURN
1210 PS=PS+1:ST(PS)=P:RETURN
2000 DIMI$(255),ST(9):POKE36879,30:PRINT"▮":LB$="                         "
2010 POKE199,1:FORI=0TO255:READI$(I):NEXT:RETURN
5000 PRINT"▮----DESENSAMBLADOR----";:FORL=1TO19:PRINTLB$;:NEXT:RETURN
5010 PRINT"▮"LB$;LB$;"                         ▮";:POKE8185,160:POKE38905,2:RETURN
5100 GOSUB5000:GOSUB5010:F7=3:C7=0:GOSUB900:PRINT"TODAS LAS DIRECCIONES ";
5110 PRINT"Y VALORES SON EN DECI-MAL.";:F7=8:GOSUB900
5120 PRINT"DIR. INICIAL =>";:F7=21:GOSUB900:PRINT"▮F2:FIN▮";:F7=8:C7=16:L7=5
5130 GOSUB950:IFT8=137THENPRINT"▮":POKE36879,27:END
5140 DI=VAL(X8$):IFDI>65530THENGOSUB900:PRINT"      ";:GOTO5130
5150 RETURN
5200 GOSUB5000:GOSUB5010:F7=20:C7=0:GOSUB900:PRINT"▮F1:PAG. ANTERIOR";
5210 F7=21:GOSUB900:PRINT"F7:PAG. SIGUIENTE";:F7=22:GOSUB900
5220 PRINT"F2:FIN▮";:RETURN
6000 DATAKBRK,HORA,,,,BORA,BASL,,KPHP,AORA,KASL,,,EORA,EASL,,LBPL,IORA,,,,CORA,C
ASL
6010 DATA,KCLC,GORA,,,,FORA,FASL,,EJSR,HAND,,,BBIT,BAND,BROL,,KPLP,AAND,KROL,,EB
IT,EAND
6020 DATAEROL,,LBMI,IAND,,,,CAND,CROL,,KSEC,GAND,,,,FAND,FROL,,KRTI,HEOR,,,,BEOR
,BLSR
6030 DATA,KPHA,AEOR,KLSR,,EJMP,EEOR,ELSR,,LBVC,IEOR,,,,CEOR,CLSR,,KCLI,GEOR,,,,F
EOR
6040 DATAFLSR,,KRTS,HADC,,,,BADC,BROR,,KPLA,AADC,KROR,,JJMP,EADC,EROR,,LBVS,IADC
6050 DATA,,,CADC,CROR,,KSEI,GADC,,,,FADC,FROR,,,HSTA,,,BSTY,BSTA,BSTX,,KDEY,,KTX
A,,ESTY
6060 DATAESTA,ESTX,,LBCC,ISTA,,,CSTY,CSTA,DSTX,,KTYA,GSTA,KTXS,,,FSTA,,,ALDY,HLD
A,ALDX
6070 DATA,BLDY,BLDA,BLDX,,KTAY,ALDA,KTAX,,ELDY,ELDA,ELDX,,LBCS,ILDA,,,CLDY,CLDA,
DLDX
6080 DATA,KCLV,GLDA,KTSX,,FLDY,FLDA,GLDX,,ACPY,HCMP,,,BCPY,BCMP,BDEC,,KINY,ACMP,
KDEX
6090 DATA,ECPY,ECMP,EDEC,,LBNE,ICMP,,,,CCMP,CDEC,,KCLD,GCMP,,,,FCMP,FDEC,,ACPX,H
SBC
7000 DATA,,BCPX,BSBC,BINC,,KINX,ASBC,KNOP,,ECPX,ESBC,EINC,,LBEQ,ISBC,,,,CSBC,CIN
C
7010 DATA,KSED,GSBC,,,,FSBC,FINC,BCON
```

Base de datos

Hoy día, el flujo de informaciones en las sociedades desarrolladas, como la nuestra, es cada vez mayor y resulta complicado para las personas asimilar toda la información que reciben desde su entorno. En pleno inicio de la «Era de la Información», son cada vez más importantes los medios que ayudan al hombre a organizar, almacenar y recuperar informaciones de toda índole y que le permiten tomar decisiones rápidas y, sobre todo, seguras por estar basadas en datos reales. Por ello, las Bases de Datos son una de las más importantes aplicaciones desarrolladas en el mundo informático. Y, por supuesto, hay algunas versiones sencillas para ordenadores domésticos, como esta que nos remite para el Commodore 64 Manuel Castell, lector de nuestra revista de Guadalajara, para, según sus propias palabras, «poder ordenar todo lo imaginable».

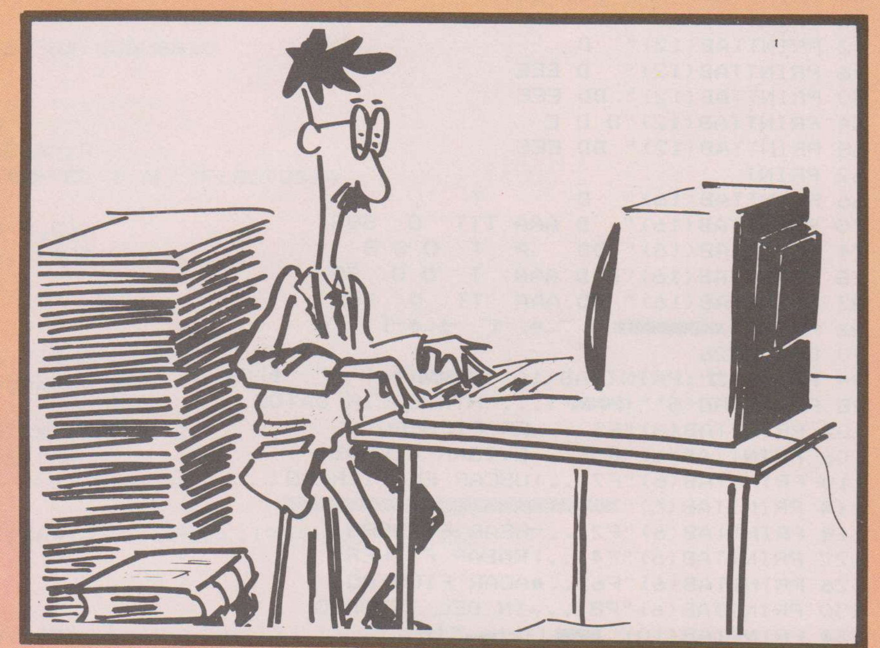
La utilización de esta base de datos es muy sencilla, pues siempre está presente el menú principal, del que se toma la opción deseada en cada caso.

Al hacer un RUN, tras teclear el programa, aparece la presentación y, seguidamente, el generador de caracteres rediseña nueve caracteres castellanos, que son:

C= n es la letra Ñ
C= m es la letra Ñ
C= a es la letra á
C= e es la letra é
C= i es la letra í
C= o es la letra ó
C= u es la letra ú
es el signo ¿
es el signo ¡
(C= es la tecla Commodore)

Las opciones que aparecen a continuación en el menú principal son:

F1. Introducir datos. Una vez creado el fichero con F2, con esta opción se presentan los campos que se han definido.
F2. Crear un fichero nuevo. Todo lo que hay en el ordenador se borra (se puede guardar un fichero anterior con F4) y se puede crear un fichero nuevo con diferente dimensión que el anterior, definiendo sus correspondientes campos.
F3. Consultar los datos que hay en el fichero. Se puede examinar todo el fichero, ficha a ficha. Con la tecla «f» se vuelve al menú principal.
F4. Grabar un fichero en el cassette.
F5. Revisar fichas. Se pueden corregir datos introducidos erróneamente en una ficha.
F6. Cargar un fichero grabado previamente con F4.
F7. Buscar en el fichero. Se puede pedir al programa que presente todas las fichas en las que aparece una determinada cadena de caracteres en cualquier campo.
F8. FIN del trabajo.



Con sólo seguir las instrucciones del programa y completar los datos que solicita, se puede crear rápidamente una útil y sencilla base de datos. Para introducir los caracteres castellanos generados basta con pulsar las teclas indicadas anteriormente. Sin embargo, aquellos que quieran resumir algo el programa pueden suprimir esta posibilidad eliminando el generador de caracteres de la siguiente manera:

Líneas	Sustituir por:
86	(Suprimir)
90	90 GOSUB 610
526 a 606	(Suprimir)

La estructura de «Base de Datos» para los que están interesados se detalla a continuación:

2-90	Presentación del programa y el autor.
94-170	Menú principal y elección de opción.
174-210	Creación de fichero.
214-254	Introducción de datos.
258-302	Examen del fichero ficha a ficha.
306-338	Revisión y rectificación de fichas.
342-406	Búsqueda de cadenas en el fichero.
410-450	Grabar fichero desde cassette.
454-518	Cargar fichero desde cassette.
526-570	Generador de caracteres ñ, Ñ, á, é, í, ó, ú, ¿, ¡.
574-606	DATAS de los caracteres ñ, Ñ, á, é, í, ó, ú, ¿, ¡.

PROGRAMAS

CONCURSO
PREMIADO CON
5.000
PESETAS

```
2 REM*********BASE DE DATOS************
6 REM           MANUEL CASTELL
10 REM********12-MAYO-1985***************
14 POKE53281,0:POKE53280,0:PRINT"  "CHR$(8)CHR$(14)
18 PRINT"B
22 PRINT"B    AAA SSS EEE
26 PRINT"BB     A S    EEE
30 PRINT"B B AAA   SS E
34 PRINT"BB   AAA SSS EEE
38 PRINT
42 PRINTTAB(12)"   D
46 PRINTTAB(12)"   D EEE
50 PRINTTAB(12)" DD EEE
54 PRINTTAB(12)"D D E
58 PRINTTAB(12)" DD EEE
62 PRINT
66 PRINTTAB(16)"   D        T
70 PRINTTAB(16)"   D AAA TTT   O   SSS
74 PRINTTAB(16)" DD    A   T   O O S
78 PRINTTAB(16)"D D AAA   T   O O   SS
82 PRINTTAB(16)" DD AAA   TT   O   SSS
86 PRINT"
90 GOSUB526
94 PRINT" ":PRINTTAB(10)"
98 PRINTTAB(6)"    F1...-NTRODUCIR DATOS
102 PRINTTAB(6)"F3...>ER FICHERO
106 PRINTTAB(6)"F5...-EVISAR FICHAS
110 PRINTTAB(6)"F7...IUSCAR EN FICHERO
114 PRINTTAB(6)"
118 PRINTTAB(6)"F2...-REAR FICHERO
122 PRINTTAB(6)"F4...IRABAR FICHERO
126 PRINTTAB(6)"F6...♥ACAR FICHERO
130 PRINTTAB(6)"F8...-IN DEL TRABAJO
134 PRINTTAB(10)"
138 GETA$:IFA$=" "THEN214:   REM F1
142 IFA$=" "THEN174:         REM F2
146 IFA$=" "THEN258:         REM F3
150 IFA$=" "THEN410:         REM F4
154 IFA$=" "THEN306:         REM F5
158 IFA$=" "THEN454:         REM F6
162 IFA$=" "THEN342:         REM F7
166 IFA$<>" "THEN138
170 END:                     REM F8
174 REM CREAR FICHERO
178 RUN182
182 INPUT"   ↑/OMBRE DEL FICHERO";N$
186 INPUT"  ↑/UMERO DE CAMPOS";K:K=K-1
190 INPUT"  ↑/UMERO DE FICHAS";N
194 DIM A$(N,K)
198 FORC=0TOK
202 PRINT" ↑/OMBRE DEL CAMPO"C;:A$(0,C)="CAMPO "+STR$(C):INPUTA$(0,C)
206 NEXTC
210 GOTO94
214 REM INTRODUCIR DATOS
218 F=F+1
222 PRINT"  -NTRDUCIR DATOS EN FICHA NUMERO"F
226 IFF>NTHENPRINT"  ←/O HAY MAS FICHAS!":F=N:GOSUB610:GOTO94
230 FORC=0TOK
234 PRINTA$(0,C):INPUTA$(F,C)
238 NEXTC
242 PRINT"   -IN DE LA FICHA NUMERO:"F
246 INPUT"  ↑♥UIERES UNA NUEVA FICHA  [S/N]";R$
250 IFR$="S"ORR$="♥"THEN214
254 GOTO94
258 REM VER FICHERO
```

```
262 FORR=1TOF
266 PRINT"   ": "N$" FICHA"R
270 FORC=0TOK
274 IFPEEK(1824)=32ANDPEEK(1864)=32ANDPEEK(1904)=32THEN286
278 GOSUB610
282 PRINT"
286 PRINT" "A$(0,C):PRINTA$(R,C)
290 NEXTC
294 GOSUB610:IFA$="F"THEN94
296 NEXTR
298 PRINT"   IN DEL FICHERO "N$:GOSUB610
302 GOTO94
306 REM REVISAR FICHERO
310 PRINT"   EVISAR FICHAS
314 INPUT"  UMERO DE LA FICHA";R
318 IFR<1 OR R>F THENPRINT"   EL 1 AL "F:GOTO314
322 FORC=0TOK
326 PRINT" "A$(0,C):PRINTA$(R,C)
330 A$=A$(R,C):INPUTA$:A$(R,C)=A$
334 NEXTC
338 GOTO94
342 REM BUSCAR
346 PRINT"    USCAR EN EL FICHERO "N$
350 INPUT"   NTRODUCIR EL NOMBRE:";R$
354 FORT=1TOF:FORC=0TOK
358 IFR$<>LEFT$(A$(T,C),LEN(R$))THEN398
362 PRINT"  ICHA NUMERO"T"DE "N$
366 PRINT"----------------------------------------
370 FORR=0TOK
374 IFPEEK(1824)=32ANDPEEK(1864)=32ANDPEEK(1904)=32THEN386
378 GOSUB610
382 PRINT"
386 REM   PRINTA$(T,R)
390 PRINT" "A$(0,R):PRINTA$(T,R)
394 NEXTR:GOSUB610
398 NEXTC,T
402 PRINT"   IN DE LA BUSQUEDA":GOSUB614
406 GOTO94
410 REM GRABAR FICHERO
414 PRINT"    RABAR FICHERO
418 OPEN1,1,1,N$:PRINT#1,N$
422 PRINT#1,N:PRINT#1,K
426 FORR=0TOF
430 FORC=0TOK
434 IFA$(R,C)=""THENA$(R,C)="@"
438 PRINT#1,A$(R,C)
442 NEXTC,R
446 PRINT#1,"FIN"
450 CLOSE1:GOTO94
454 REM CARGAR FICHERO
458 PRINT"    ARGAR FICHERO
462 RUN466
466 OPEN1,1,0:INPUT#1,N$
470 INPUT#1,N:INPUT#1,K
474 DIMA$(N,K)
478 FORR=0TON
482 FORC=0TOK
486 INPUT#1,A$
490 IFA$="FIN"THEN502
494 A$(R,C)=A$
498 NEXTC,R
502 CLOSE1
506 PRINT"   ICHERO: "N$
510 PRINT"  UMERO MAXIMO DE CAMPOS"K+1
514 PRINT"  UMERO MAXIMO DE FICHAS"N
```

```
518 PRINT"▮/UMERO DE FICHAS CON DATOS"R-1:F=R-1
522 GOSUB610:GOTO94
526 REM GENERADOR DE CARACTERES
530 POKE52,48:POKE56,48
534 POKE56334,PEEK(56334)AND254
538 POKE1,PEEK(1)AND251
542 FORR=0TO2047:POKER+12288,PEEK(R+55296):NEXT
546 POKE1,PEEK(1)OR4
550 POKE53272,(PEEK(53272)AND240)+12
554 FORR=0TO8:READC
558 FORA=0TO7:READB:POKEC+A,B:NEXTA
562 NEXTR
566 POKE56334,PEEK(56334)OR1
570 RETURN
574 DATA13112,60,0,102,118,126,110,102,0
578 DATA13136,60,0,124,102,102,102,102,0
582 DATA13184,12,24,60,6,62,102,62,0
586 DATA13192,12,24,60,102,126,96,60,0
590 DATA13072,12,24,0,56,24,24,60,0
594 DATA13256,12,24,60,102,102,102,60,0
598 DATA13248,12,24,102,102,102,102,62,0
602 DATA12536,24,0,0,24,24,24,24,0
606 DATA12528,24,0,24,48,96,102,60,0
610 POKE781,23:POKE782,0:SYS 65520:PRINT"-----PULSA UNA TECLA.
614 POKE180,0:A$=""
618 GETA$:IFA$=""THEN618
622 RETURN
```

CODIGOS DE CONTROL PARA EL VIC-20 Y EL C-64

Colores del VIC-20 y del 64

Cómo se ve	Cómo se teclea	Efecto conseguido
	Ctrl+1	Negro
	Ctrl+2	Blanco
	Ctrl+3	Rojo
	Ctrl+4	Cian
	Ctrl+5	Púrpura
	Ctrl+6	Verde
	Ctrl+7	Azul
	Ctrl+8	Amarillo

Colores del 64 solamente

Cómo se ve	Cómo se teclea	Efecto conseguido
	Cbm+1	Naranja
	Cbm+2	Marrón
	Cbm+3	Rosa
	Cbm+4	Gris oscuro
	Cbm+5	Gris medio
	Cbm+6	Verde claro
	Cbm+7	Azul claro
	Cbm+8	Gris claro

Códigos de cursor y control

Cómo se ve	Cómo se teclea	Efecto conseguido
	Home	Cursor a casa
	Shift+home	Limpia pantalla
	Crsr	Cursor derecha
	Shift+crsr	Cursor izquierda
	Crsr	Cursor abajo
	Shift+crsr	Cursor arriba
	Ctrl+9	Carácter inverso
	Ctrl+0	Carácter normal
	Del	Borrar
	Shift+del	Insertar

Teclas de función

Cómo se ve	Cómo se teclea
	F1
	F2=Shift+F1
	F3
	F4=Shift+F3
	F5
	F6=Shift+F5
	F7
	F8=Shift+F7



LE OFRECE LOS MEJORES LIBROS PARA SU ORDENADOR



P.V.P. 750 PTAS.
(IVA INCLUIDO)
Descubre los misterios de la programación de una forma sencilla, con ejemplos, programas y organigramas.
(110 páginas, tamaño 13,5 x 21)



P.V.P. 800 PTAS.
(IVA INCLUIDO)
Con utilidades, juegos explosivos y gráficos dinámicos que lleva al BASIC hasta el mejor aprovechamiento de sus posibilidades.
(200 páginas, tamaño 15,5 x 21,5).



P.V.P. 750 PTAS.
(IVA INCLUIDO)
Un libro especialmente dedicado a los que se inician por vez primera en el mundo del Spectrum.
(100 páginas, tamaño 13,5 x 21).



P.V.P. 800 PTAS.
(IVA INCLUIDO)
Una inestimable ayuda que complementará la que proporciona el manual del ordenador.
(108 páginas tamaño 13,5 x 21,5).



P.V.P. 900 PTAS.
(IVA INCLUIDO)
Un compendio de los programas más diversos con los que podrá aprender jugando las importantnes características del BASIC.
(258 páginas, tamaño 15,5 x 21,5).



P.V.P. 800 PTAS.
(IVA INCLUIDO)
Muestra una visión más completa del correcto funcionamiento del juego de instrucciones del C-64.
(108 páginas, tamaño 13,5 x 21,5).

CUPON DE PEDIDO

enviar a:

infodis, s.a.

C/BRAVO MURILLO, 377
28020 MADRID

COPIE O RECORTE ESTE BOLETIN DE PEDIDO.

DESEO RECIBIR LOS SIGUIENTES TITULOS:

15 HORAS CON EL SPECTRUM (P.V.P. 750) ☐
LOS MEJORES PROGRAMAS PARA EL ZX SPECTRUM (P.V.P. 900) ☐
LOS MEJORES PROGRAMAS PARA EL COMMODORE 64 (P.V.P. 800) ☐
EL 64 MAS ALLA DEL MANUAL I (P.V.P. 800) ☐
EL 64 MAS ALLA DEL MANUAL II (P.V.P. 800) ☐
(más 100 ptas. de gastos de envío).

El importe lo abonaré POR CHEQUE ☐ CONTRA REEMBOLSO ☐ CON MI TARJETA DE CREDITO ☐ American Express ☐ Visa ☐ Interbank ☐

Número de mi tarjeta: ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

NOMBRE ______________________________

CALLE ______________________________

CIUDAD ______________________________

PROVINCIA ______________________ C. P. __________

commodore Magazine SERVICIO

DISPONEMOS DE TAPAS ESPECIALES PARA SUS EJEMPLARES DE ZX
(sin necesidad de encuadernación)



Núm. 2 - 250 Ptas.
CBM 64 en profundidad/Superbase 64: el ordenador que archiva/Juegos, trucos y aplicaciones.



Núm. 3 - 250 Ptas.
Magic Desk, el despacho en casa/Herramientas para el programado/Interfaces para todos.



Núm. 4 - 250 Ptas.
El 64 transportable revisado a fondo/Interface RS 232 para el VIC-20/Juegos/El fútbol-silla en su salón.



Núm. 5 - 250 Ptas.
Programas, juegos y concurso/Londres: Quinta feria de Commodore/Basic, versión 4.75.



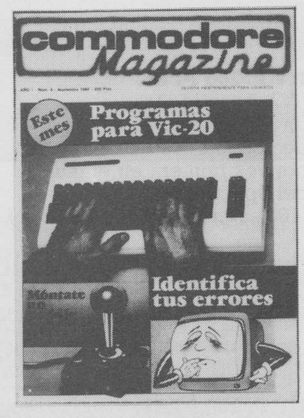
Núm. 6 - 250 Ptas.
El misterio del Basic/Lápices ópticos para todos/Concurso, juegos, aplicaciones.



Núm. 7 - 250 Ptas.
El ordenador virtuoso. Musi-Calc. Programa monitor para el 64. Lápices ópticos. Ampliación de memoria para Vic-20.



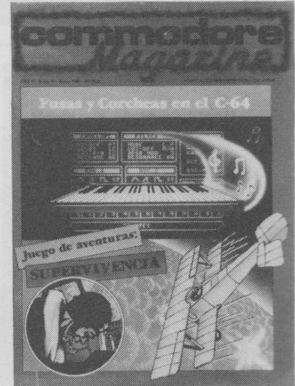
Núm. 8 - 250 Ptas.
Joystick y Paddle para todos. Misterio del BASIC. EL LOGO. Cálculo financiero. Programas.



Núm. 9 - 250 Ptas.
Conversión de programas del Vic-20 al C-64. Móntale un paddle. Identifica tus errores. Software comentado.



Núm. 10 - 250 Ptas.
Koala Pad: La potencia de un paquete gráfico. Trucos. El FORTH. Software comentado. El LOGO.



Núm. 11 - 250 Ptas.
Music-64. Supervivencia (1.ª parte). Cómo guarda el diskette la información. Sintetizador-64. El Forth (1.ª parte).



Núm. 12 - 250 Ptas.
Commodore-16 por dentro y por fuera. Sprites: los alegres duendecillos (1.ª parte). Supervivencia (1.ª parte). El Forth (y 3.ª parte).



Núm. 13 - 250 Ptas.
Análisis: programas de ajedrez. Los Cazafantasmas. 64 Vic en el espacio. La impresora que dibuja. Interface paralelo.

DE EJEMPLARES ATRASADOS

Complete su colección de COMMODORE MAGAZINE

A continuación le resumimos el contenido de los ejemplares aparecidos hasta ahora.



Núm. 14 - 250 Ptas.
Sprites; cómo entenderse con los duendes. Pilot: un lenguaje de alto nivel. Guía de Software para C-64.



Núm. 15 - 250 Ptas.
Síntesis de voz: su ordenador tiene la palaba. Pilot: un lenguaje de alto nivel (2.ª parte). Guía de software para C-64 (2.ª parte).



Núm. 16 - 250 Ptas.
Análisis de simuladores: vuela con tu C-64. Contabilidad para pequeños negocios. Cómo acelerar la ejecución de gráficos en BASIC. Submarino Commander. Pilot: un lenguaje de alto nivel (3.ª Parte).



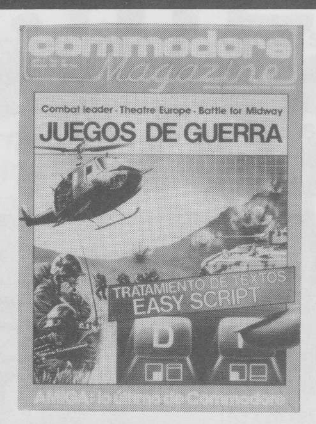
Núm. 17 - 250 Ptas.
Una lección de anatomía: los microordenadores por dentro. Bruce Lee: la furia oriental en el C-64. Quick Data Drive. Colossus Chess: un coloso del ajedrez.



Núm. 18 - 250 Ptas.
Practicalc: todo el poder de una hoja electrónica. Pascal (1.ª parte). Programas: juegos y aplicaciones veraniegos. ¡Canasta!: dos ases del baloncesto para el C-64.



Núm. 19 - 250 Ptas.
Análisis de cuatro impresoras. Simulación: pequeños mundos en su ordenador. Pascal (2.ª parte). Entombed y The Staff of Karnath: aventuras gráficas y mucha acción.



Núm. 20 - 250 Ptas.
Juegos de Guerra: Combat leader, Theatre Europe, Battle for Midway. Tratamiento de textos Easy Script. Amiga: lo último de Commodore. Libros, juegos y aplicaciones.



Núm. 21 - 250 Ptas.
Video-Digitizer: visión artificial para Commodore. Seikosha GP 700 VC: una impresora a todo color y con capacidades gráficas. Sprites multicolores. El nuevo C-128.



Núm. 22 - 300 Ptas.
Programas lightning: gráficos profesionales a tu alcance. Montaje: un interruptor programable para el C-64. Sprites múltiples. Cómo graba los datos el Datassette.



Núm. 23 - 300 Ptas.
Sinfonías en Chip: síntesis de sonido. Sprites en movimiento. Paisajes fractales en tu Commodore. Código máquina. Sight and sound: cuatro maestros de la música. Galería de Software: los mejores programas analizados en profundidad.

Corte y envíe este cupón a: COMMODORE MAGAZINE
Bravo Murillo, 377 - Tel. 7337969 - 28020-MADRID

SERVICIO DE EJEMPLARES ATRASADOS

Ruego me envien los siguientes ejemplares atrasados de COMMODORE MAGAZINE
____

El importe lo abonare:
Contra reembolso ☐ Adjunto Cheque ☐ Con mi tarjeta de credito ☐
American Express ☐ Visa ☐ Interbank ☐ Fecha de caducidad ______

Numero de mi tarjeta ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

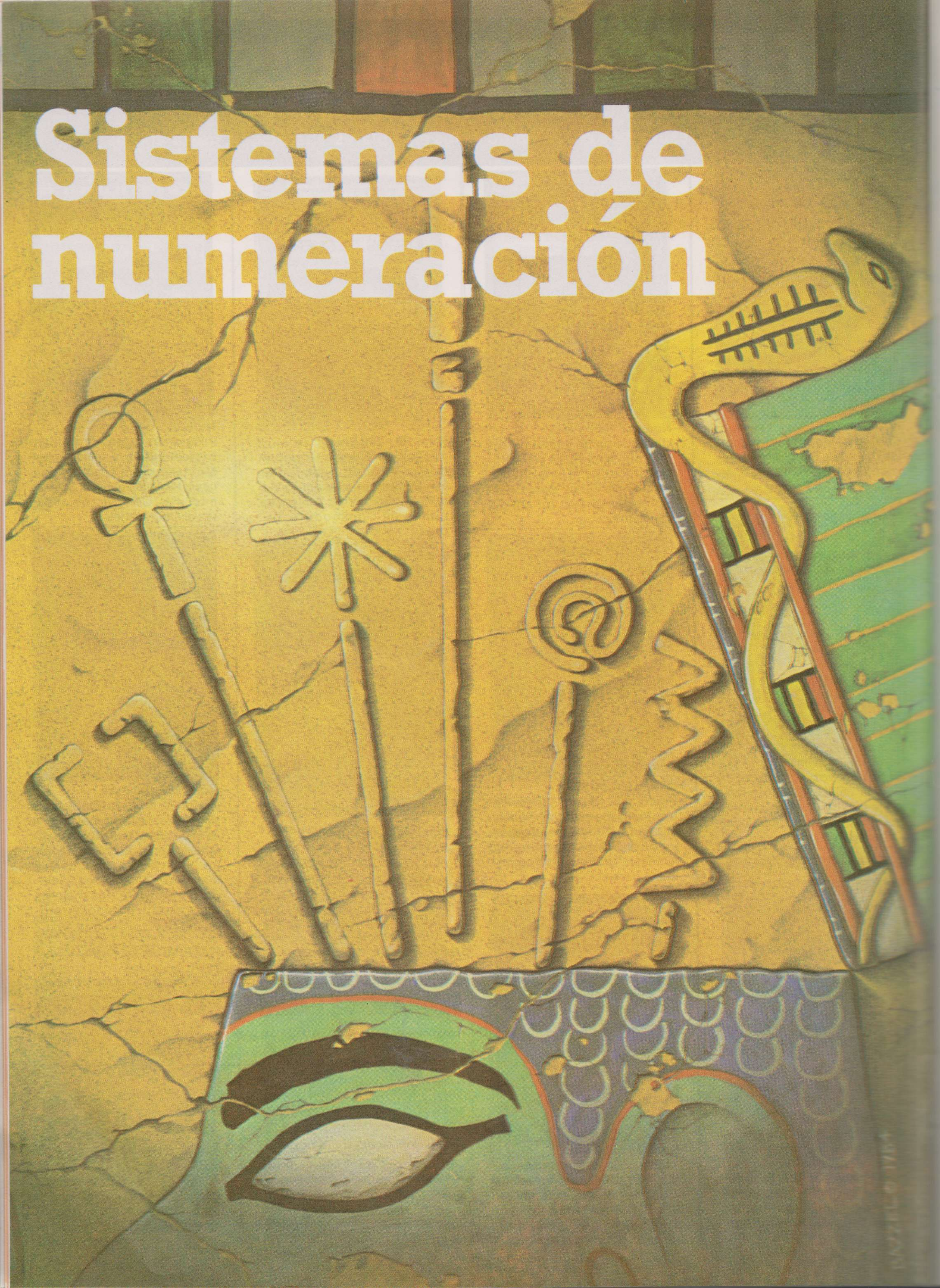
NOMBRE ______________________________

DIRECCION ______________________________

CIUDAD ______________________________ D.P. ______

PROVINCIA ______________________________

Sistemas de numeración



El hombre adoptó el sistema de numeración decimal por el simple hecho de estar dotado de 10 dedos en las manos. El ordenador se comunica en sistema binario por transmitir con impulsos eléctricos que simulan el 0 y el 1, y las memorias de los ordenadores trabajan en sistema hexadecimal para simplificar al hombre la utilización del sistema binario.

Sistema hexadecimal

Entre todos los sistemas de numeración, el hexadecimal es el de mayor interés para su empleo con ordenadores. Por una parte es fácil imaginar lo complicado que resultaría la programación y el uso de rutinas en lenguaje máquina si hubiera que escribirlas en el sistema binario. Por otro lado podemos observar que con sólo dos cifras en hexadecimal se puede escribir cualquier número entre 0 y 255 (casualmente el valor más alto que puede tomar un octeto) y con 4 cifras se pueden escribir todos los números que pueden almacenarse en dos octetos (o uno en ordenadores de 16 bits). En un octeto podemos escribir números desde el 0 al 255, por lo que si tomamos dos octetos, podemos escribir $255^2 = 65.535$ números además del 0. Quizá lo más práctico del sistema hexadecimal es que cada una de sus cifras equivale a un bloque de 4 cifras en sistema binario, como veremos. Así podemos dividir un octeto en dos bloques de 4 cifras y cada bloque queda perfectamente identificado por una cifra hexadecimal.

COMO PASAR UN NUMERO DEL SISTEMA HEXADECIMAL AL DECIMAL

El sistema de numeración hexadecimal utiliza 16 cifras. Como nosotros sólo manejamos 10 cifras (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9), tendremos que dar nombre a las otras 6 cifras necesarias. Así, utilizaremos las letras A, B, C, D, E y F como nuevas cifras cuyos valores en el sistema decimal son, respectivamente, 10, 11, 12, 13, 14 y 15.

Recordemos que en el sistema decimal, el número **abcde** es igual a:

$$abcde = a \cdot 10^4 + b \cdot 10^3 + c \cdot 10^2 + d \cdot 10 + e$$

Si escribimos un número cuyas cifras son **abcde** en sistema hexadecimal y queremos saber cuál es su correspondiente valor en el sistema decimal, bastará hacer lo mismo que en (2), sólo que ahora utilizaremos 16 como base en lugar de 10:

$$abcde = a \cdot 16^4 + b \cdot 16^3 + c \cdot 16^2 + d \cdot 16 + e$$

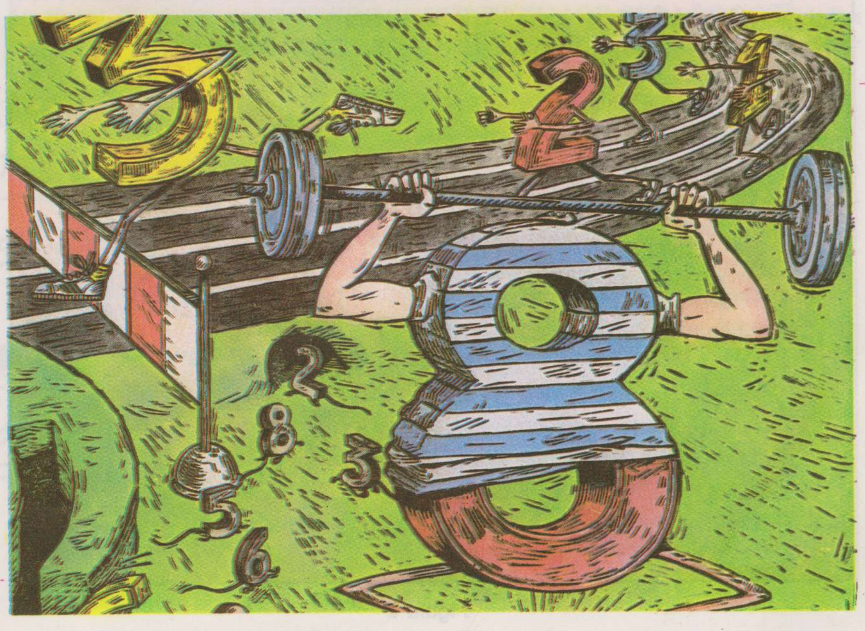
En general, el equivalente en el sistema decimal de un número hexadecimal de k + 1 cifras es:

$$a_k\ a_{k-1}\ a_{k-2} \dots a_2\ a_1\ a_0 = a_k \cdot 16^k + a_{k-1} \cdot 16^{k-1} + a_{k-2} \cdot 16^{k-2} + \dots + a_2 \cdot 16^2 + a_1 \cdot 16^1 + a_0 \cdot 16^0$$

Vamos, por ejemplo, a pasar el número **A10BF** escrito en el sistema decimal, al correspondiente en el decimal (recordemos que A = 10, B = 11, C = 12, D = 13, E = 14, F = 15):

$$A10BF_{(16} = A \cdot 16^4 + 1 \cdot 16^3 + 0 \cdot 16^2 + B \cdot 16 + F = 10 \cdot 16^4 + 1 \cdot 16^3 + 0 \cdot 16^2 + 11 \cdot 16 + 15 = 10 \cdot 65.536 + 1 \cdot 4.096 + 11 \cdot 16 + 15 = 659.647_{(10}$$

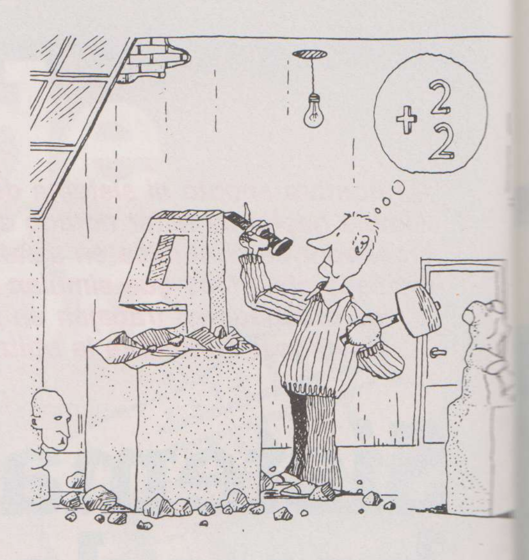
Con la siguiente rutina se pueden pasar números en base 16 al sistema decimal: **(Ver figura 1)**



COMO PASAR UN NUMERO DEL SISTEMA DECIMAL AL HEXADECIMAL.

De forma análoga a como vimos al pasar un número en base 10 al correspondiente en base 2, para pasar un número A del sistema decimal al hexadecimal, bastará hacer la división entera de A por 16 y conservar el resto. A continuación se hace lo mismo con el cociente resultante, y así sucesivamente hasta que tengamos un resto menor que 16. Veámoslo con un ejemplo: Pasemos el número 123456 (decimal) al sistema hexadecimal:

```
123456 |16
 114    7716 |16
  25     131   482 |16
  96      36    02   30 |16
   0       4     2   14    1
   =       =     =    =    =
   a4      a3    a2   a1   a0
```



Por tanto:

$$123456_{(10} = 1E240_{(16}$$

La dirección más alta que admite *BASIC* es 65535. En hexadecimal:

$$65535_{(10} = FFFF_{(16}$$

El valor más alto que puede tomar un octeto es 255, que en base 16 es FF.

Pensemos en un octeto de 16 *bits* y veamos algunas equivalencias:

$1111 \cdot 1111 \cdot 1111 \cdot 1111_{(2} = 65535_{(10} = FFFF_{(16}$
$0000 \cdot 1111 \cdot 1111 \cdot 1111_{(2} = 4095_{(10} = FFF_{(16}$
$0000 \cdot 0000 \cdot 1111 \cdot 1111_{(2} = 255_{(10} = FF_{(16}$
$0000 \cdot 0000 \cdot 0000 \cdot 1111_{(2} = 15_{(10} = F_{(16}$
$1000 \cdot 0000 \cdot 0000 \cdot 0000_{(2} = 32768_{(10} = 8000_{(2}$
$0000 \cdot 1000 \cdot 0000 \cdot 0000_{(2} = 2048_{(10} = 800_{(16}$
$0000 \cdot 0000 \cdot 1000 \cdot 0000_{(2} = 128_{(10} = 80_{(16}$
$0000 \cdot 0000 \cdot 0000 \cdot 1000_{(2} = 8_{(10} = 8_{(16}$

Observamos que podemos dividir un octeto de 16 *bits* en 4 bloques de 4 *bits*. El valor más alto de cada bloque es F ($15_{(10}$). Por ejemplo, el número binario (es decir, el estado real del octeto de 16 bits) correspondiente al hexadecimal ABCD es:

$$A_{(16} = 10_{(10} = 1010_{(2}$$
$$B_{(16} = 11_{(10} = 1011_{(2}$$
$$C_{(16} = 12_{(10} = 1100_{(2}$$
$$D_{(16} = 13_{(10} = 1101_{(2}$$

con lo que:

$$ABCD_{(16} = 1010 \cdot 1011 \cdot 1101 \cdot 1101_{(2}$$

La siguiente rutina sirve para pasar un número decimal a hexadecimal: **(Ver figura 2)**

```
10000 A$="0123456789ABCDEF"
10010 INPUT"NUMERO HEXADECIMAL";H$
10020 D=0
10030 FOR I=1 TO LEN(H$)
10040 B$=MID$(H$,I,1)
10050 FOR J=1 TO 16
10060 C$=MID$(A$,J,1)
10070 IF B$=C$ THEN 10090
10080 NEXT J
10090 D=D+(J-1)*16↑(LEN(H$)-I)
10100 NEXT I
10110 PRINT H$;" EN DECIMAL ES:";D
10120 GOTO 10010
```

Figura 1

```
20000 A$="0123456789ABCDEF"
20010 INPUT "NUMERO DECIMAL";D
20020 H$=""
20030 K$=""
20040 D1=D
20050 D1=INT(D1)/16
20060 B=16*(D1-INT(D1))
20070 IF B=INT(B) THEN 20110
20080 B1=B-INT(B)
20090 B=INT(B)
20100 IF B1>0.5 THEN B=B+1
20110 K$=K$+MID$(A$.B+1,1)
20120 IF D1>=1 THEN 20050
20130 FOR I=LEN(K$) TO 1 STEP-1
20140 H$=H$+MID$(K$,I,1)
20150 NEXT I
20160 PRINT D;"EN HEXADECIMAL ES:";H$
20170 GOTO 20010
```

Figura 2

Sistema binario

EL ordenador maneja la información mediante pequeñas unidades de memoria que llamamos OCTETOS. Por ejemplo, el ordenador almacena la palabra CUENTA en seis de dichas unidades, de forma que en cada octeto se almacena una letra. En realidad, lo que se almacena en los octetos no son las letras C, U, E, N, T y A, sino números que sirven para identificar dichas letras. Así, en lugar de la letra C, el ordenador almacena el número 67, en vez de la letra U, el número 85, y así sucesivamente.

Cada octeto está formado por ocho unidades aún más pequeñas (en los ordenadores domésticos) de memoria llamadas BITS. Cada bit puede tener dos únicos valores: cero o uno. Esto equivale a suponer que cada octeto está compuesto de ocho bombillas. Si por una bombilla pasa corriente (está encendida), le corresponde un uno, y si no pasa corriente (está apagada), le corresponde un cero.

Así el ordenador sólo puede manejar la información comprobando si los bits correspondientes están a cero o a uno.

La operación más elemental que puede hacer el ordenador es la suma de dos octetos. Si en un octeto está el número 3, y en otro el número 4, ambos números estarán escritos mediante ceros y unos, y su suma (7) será otro número formado por ceros y unos.

Así observamos que el ordenador no puede, como nosotros, trabajar en base 10 (en esta base tenemos diez dígitos: 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9), sino que debe trabajar en una base de sólo dos dígitos (0 y 1), es decir, en base 2. Esta es la razón por la que es tan interesante el estudio del sistema de numeración en base 2. Por ejemplo, el ordenador divide la pantalla del televisor o monitor en PIXELS. Un pixel corresponde a un punto y es la figura más pequeña que puede dibujar en la pantalla. A cada pixel le corresponde un bit de un octeto particular, de forma que al escribir o dibujar en la pantalla, el ordenador pondrá un punto (del color de la tinta) si el bit correspondiente está a 1 y no pondrá nada si el bit está a 0.

COMO PASAR UN NUMERO DEL SISTEMA BINARIO AL DECIMAL.

Podemos leer el número 74.321 (en base 10) de la siguiente forma:

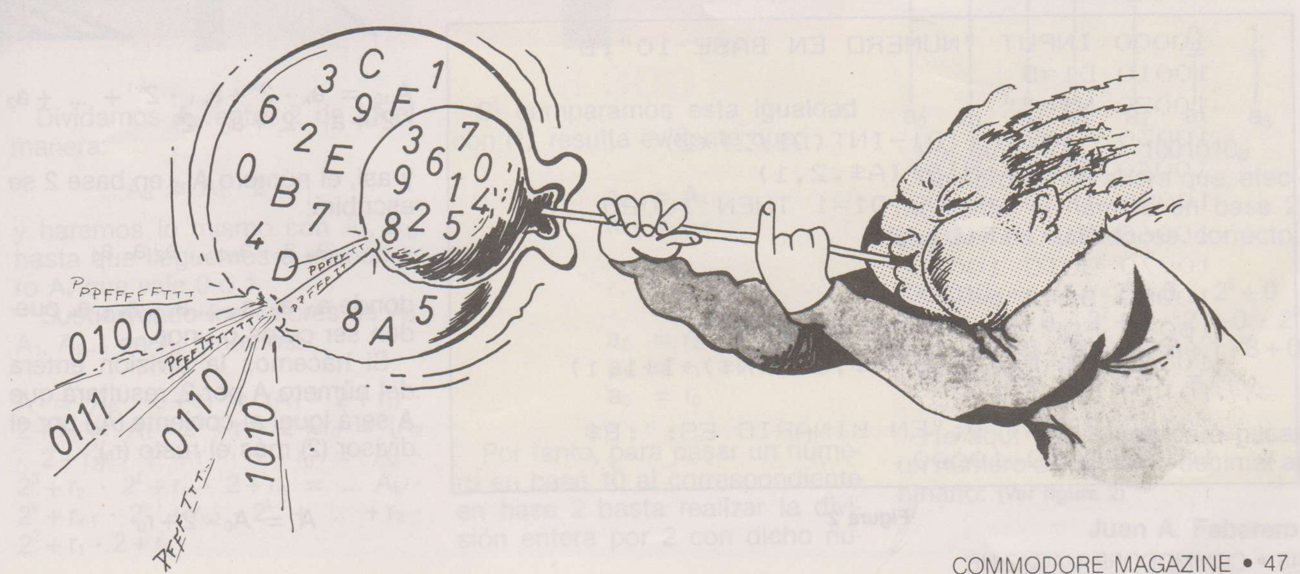
74.321 es un número con una unidad, dos decenas, tres centenas, cuatro unidades de millar y siete decenas de millar. Así, otra forma de escribir este número es la siguiente:

$74.321 = 7 \cdot 10.000 + 4 \cdot 1.000 + 3 \cdot 100 + 2 \cdot 10 + 1 = 7 \cdot 10^4 + 4 \cdot 10^3 + 3 \cdot 10^2 + 2 \cdot 10^1 + 1 \cdot 10^0$

Naturalmente es más cómodo escribir 74.321 y decir setenta y cuatro mil trescientos veintiuno.

Para pasar un número en base 2 al correspondiente número en base 10, basta por tanto hacer lo mismo que hemos hecho con 74.321, sólo que ahora la base es 2 en lugar de 10 y los únicos dígitos disponibles son 0 y 1. Así, para escribir el número $a_k\ a_{k-1}\ a_{k-2} \ldots a_2\ a_1\ a_{0\,(2}$ en base 2 (donde $a_k, a_{k-1}, \ldots a_1, a_0$ pueden ser 0 ó 1) en el sistema de base 10, tendremos que hacer:

$a_k\, a_{k-1}\, a_{k-2} \ldots a_2\, a_1\, a_{0\,(2} = a_k \cdot 2^k + a_{k-1} \cdot 2^{k-1} + a_{k-2} \cdot 2^{k-2} + \ldots + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0$



Si, por ejemplo, queremos hallar el número en base 10 correspondiente a $1001110101_{(2}$, tendremos que hacer:

$1001110101_{(2} = 1 \cdot 2^9 + 0 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1 \cdot 512 + 0 \cdot 256 + 0 \cdot 128 + 1 \cdot 64 + 1 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 = 629_{(10}$

Así vemos que el número más alto que puede tomar un octeto (sus 8 bits a 1) es:

$11111111_{(2} = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255_{(10}$

Por tanto, podemos escribir 256 números diferentes en un octero (de 0 a 255), lo que equivale a decir que podemos identificar 256 caracteres distintos (ésta es la razón por la que el Código ASCII tiene 256 caracteres exactamente).

La siguiente rutina sirve para convertir un número dado en sistema binario al correspondiente número en el sistema decimal. (Ver figura 1)

COMO PASAR UN NUMERO DEL SISTEMA DECIMAL AL BINARIO.

Supongamos que queremos escribir un número $A_{(10}$ en el sistema binario. Deberemos escribir el número A en la forma:

$A_{(10} = a_k \cdot 2^k + a_{k-1} \cdot 2^{k-1} + \ldots + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0$

y así, el número $A_{(10}$ en base 2 se escribirá:

$a_k \; a_{k-1} \; a_{k-2} \ldots a_2 \; a_1 \; a_0$

donde a_0, a_1, $a_2 \ldots a_{k-2}$, a_{k-1}, a_k pueden ser ceros o unos.

Si hacemos la división entera del número A por 2, resultará que A será igual al cociente (A_0) por el divisor (2) más el resto (r_0):

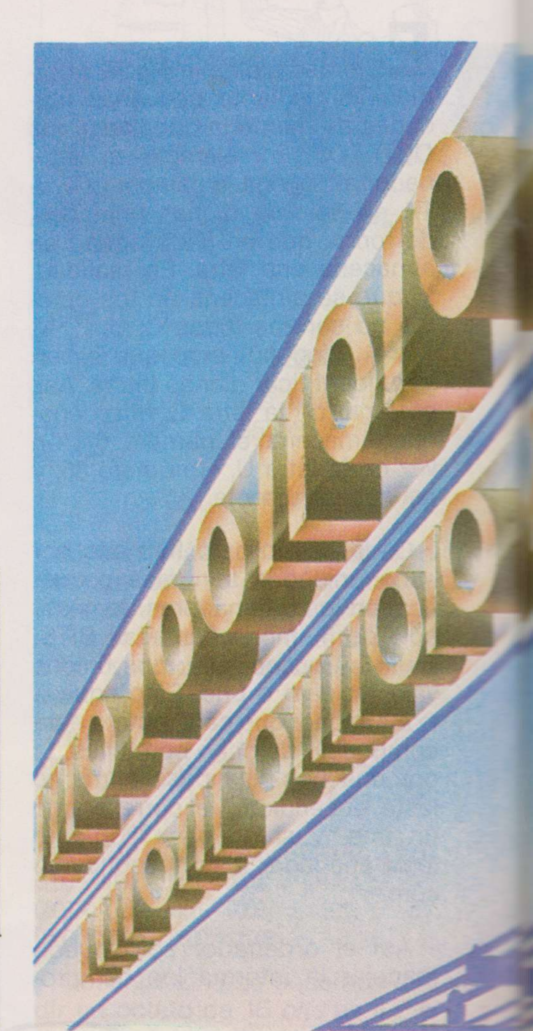
$$A = A_0 \cdot 2 + r_0$$

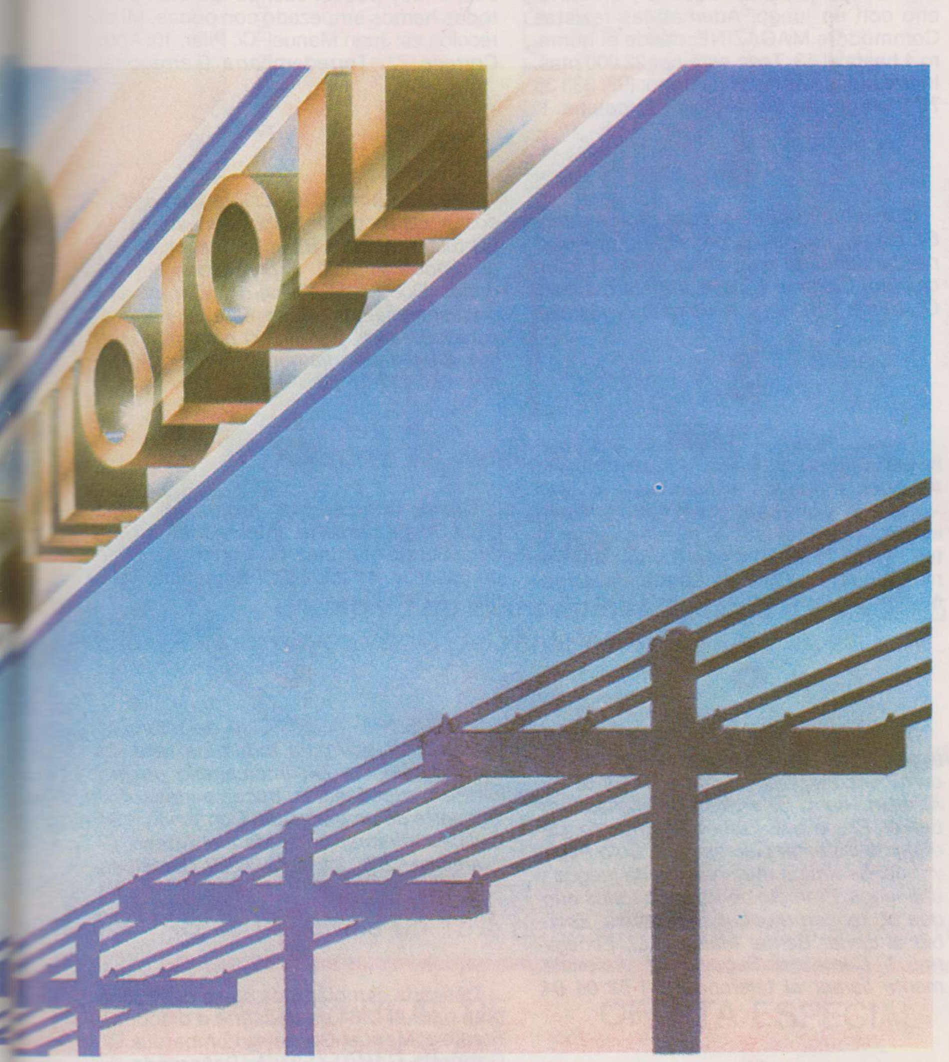
```
10000 INPUT "NUMERO EN BASE 2";B$
10010 D=0
10020 FOR I=1 TO LEN(B$)
10030 A$=MID$(B$,I,1)
10040 A=VAL(A$)
10050 D=D+A*2↑(LEN(B$)-I)
10060 NEXT I
10070 PRINT B$;" EN DECIMAL ES:";D
10080 GOTO 10000
```

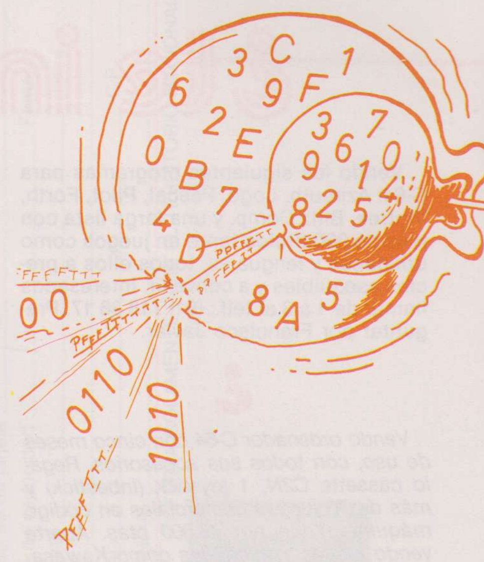
Figura 1

```
10000 INPUT "NUMERO EN BASE 10";D
10010 D1=D
10020 N$=""
10030 A$=STR$(D1-INT(D1/2)*2)
10040 N$=N$+MID$(A$,2,1)
10050 IF D1=0 OR D1=1 THEN 10080
10060 D1=INT(D1/2)
10070 GOTO 10030
10080 B$=""
10090 FOR I=1 TO LEN(N$)
10100 B$=B$+MID$(N$,LEN(N$)-I+1,1)
10110 NEXT I
10120 PRINT D;"EN BINARIO ES:";B$
10130 GOTO 10000
```

Figura 2







Dividamos A_0 entre 2 de igual manera:

$$A_0 = A_1 \cdot 2 + r_1$$

y haremos lo mismo con A_1, A_2, hasta que lleguemos a un número A_k que vale 0 ó 1.

Sustituyendo los valores de A_0, A_1, A_2,... en A, tenemos:

$$A = A_0 \cdot 2 + r_0 = (A_1 \cdot 2 + r_1) \cdot 2 + r_0 = A_1 \cdot 2^2 + r_1 \cdot 2 + r_0 = (A_2 \cdot 2 + r_2) \cdot 2^2 + r_1 \cdot 2 + r_0 = A_2 \cdot 2^3 + r_2 \cdot 2^2 + r_1 \cdot 2 + r_0 = \ldots A_k \cdot 2^k + r_{k-1} \cdot 2^{k-1} + r_{k-2} \cdot 2^{k-2} + \ldots + r_2 \cdot 2^2 + r_1 \cdot 2 + r_0$$

Si comparamos esta igualdad con (1), resulta evidente que:

$$a_k = A_k$$
$$a_{k-1} = r_{k-1}$$
.
.
.
$$a_2 = r_2$$
$$a_1 = r_1$$
$$a_0 = r_0$$

Por tanto, para pasar un número en base 10 al correspondiente en base 2 basta realizar la división entera por 2 con dicho número. El resto será a_0. Luego se vuelve a hacer la división con el cociente resultante. El resto de esta segunda división será a_1. Se repite este proceso hasta que el cociente sea igual a 0 ó a 1, que será a_k.

Veamos un ejemplo: vamos a pasar el número $74_{(10}$ al correspondiente número en base 2:

```
74 |2
14  37 |2
 0  17  18 |2
     1   0   9 |2
             1   4 |2
                 0   2 |2
                     0   1
a0  a1  a2  a3  a4  a5  a6
```

Por tanto: $74_{(10} = 1001010_{(2}$

Comprobemos ahora que, efectivamente, el número en base 2 que hemos hallado es correcto:

$$1001010_{(2} = 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 1 \cdot 64 + 0 \cdot 32 + 0 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 2 + 0 = 64 + 8 + 2 = 74_{(10}$$

He aquí una rutina para pasar un número del sistema decimal al binario: **(Ver figura 2)**

Juan A. Feberero

¿te interesa?

Vendo los siguientes programas para C-64: Azimuth, Logo, Pascal, Pilot, Forth, Victree, Blitz Comp. y una larga lista con más de 300 títulos tanto en juegos como utilidades y lenguajes, todos ellos a precios asequibles y a convenir. Interesados llamar de 1 a 3 al telf.: (91) 718 36 17. Preguntar por Francisco Javier.



Vendo ordenador C-64 con cinco meses de uso, con todos sus accesorios. Regalo cassette C2N, 1 joystick (Inbestick) y más de 70 juegos comerciales en código máquina. Todo por 45.000 ptas. Aparte vendo juegos y utilidades como Kawasaki Ritmo Roker, Simon's Basic I-II, etc. Mandaré mi lista. Dirigirse a: Julio Belmonte Tornel. Avd. de la Fama, 17. Edificio Rodas. 30006 Murcia.



Vendo Vic-20 con un cartucho de ajedrez, varios programas, un curso de Basic del VIC, una guía del ordenador, un libro de programas y varias revistas. Todo por 15.000 ptas. Ponerse en contacto con: José Pedro Fernández Fernández. C/. Relatores, 3-3.°D. 47011 Valladolid. Telf.: (983) 26 61 14.



Club de Commodore 64, B.A.S.A., intercambia todo tipo de programas, sobre todo en diskette. Para contactar enviar lista a: Rafael Andrés, C/. Anselm Clavé, 53. Olesa de Montserrat (Barna). O llamar al (93) 778 06 77.



Vendo C-64 con 1541 Flash y Reset, más unidad de discos e impresora Seikosha GP-550A. Cartuchos Simon's Basic y Hesmon. Regalo más de 100.000 ptas. en programas (tengo más de 1.000 títulos de todos los tipos). Todo en perfecto estado por 150.000 ptas. Llamar a David al telf.: (91) 734 11 03. Madrid.



Vendo por cambio de ordenador: CBM-64, unidad de disco VC-1541 de Commodore y los siguientes programas: Superbase/64, Calc Result, Easy Script, Logo, Pilot, Simon's Basic, Master 64, Peet Speed, Oxford Pascal, Koala Painter y 300 juegos y utilidades más. Todo por 200.000 ptas. (valorado en más de 300.000 ptas.). También lo vendo por separado. Interesados preguntar por Toni en el telf.: 28 43 37, de 10 h. a 13 h. Palma de Mallorca.

Vendo ordenador Commodore VIC-20; con cassette ANITECH VK-160, en perfecto estado, un cursor de introducción al BASIC, el manual del usuario y un cartucho con un juego. Además las revistas Commodore MAGAZINE, desde el número 4 hasta el 13. Todo esto por 22.000 ptas. Interesados llamar al teléfono: (93) 333 35 76 y preguntar por Néstor. Barcelona.



Cambio programas para Commodore 64, trucos, utilidades, etc. Poseo una buena cantidad de programas para el intercambio. Dirigirse a: José Ute Marti Albert. C/. Carrero Blanco, 1. Belgida (Valencia).



Vendo VIC-20 + Datassette + primera parte curso de Basic + Cartucho Vic Avenger + Juegos + revistas + cables + manual y cintas vírgenes todo en buen estado. Los interesados pueden llamar al telf.: (973) 38 30 30 ó escribir a: Toni Farras Dique. C/. Ctra. de Lérida s/n. Organa (Lérida).



Cambio programas para Commodore 64 a chicos de toda España. Tengo títulos como: Jet set Willy, Zaxxon, Tales of the Arabian Night, Raid over Moscow, Ghostbuster, Bruce Lee, Fútbol, Baloncesto, Decathlon, Summer Games I y II, Solo Flight y muchos títulos muy buenos de juegos y utilidades. Prometo contestar a todos aunque no tengan muchos programas. Escribid a: Javier Bernal Malillos. C/. Echegaray, 1. Cantalejo (Segovia). O preferiblemente llamar al teléfono (911) 52 04 04.



Tengo un ordenador Commodore 64 con *cassette* y me gustaría intercambiar juegos y utilidades. Tengo más de 300 títulos y la mayoría de ellos son muy buenos. Dirigirse a: Francisco Marcos Ruiz Franquin. Apartado de Correos 108. La Cuesta (Tenerife). Contestaré a todas las cartas recibidas.



Si te gustan los juegos y utilidades para tu CBM-64, mándame rápidamente una carta con tu dirección, teléfono, edad, y una lista tuya. Mi dirección es: Juan Manuel. Apto. Correos n.° 71 de Torredembarra (Tarragona).

Atención!!! Si tienes ganas de cambiar programas para CBM-64, escríbeme. Tengo muchos y muy buenos. Si tienes pocos, o muy pocos, escribe también pues todos hemos empezado con pocos. Mi dirección es: Juan Manuel. C/. Pilar, 10. Apto. Correos 71. Torredembarra (Tarragona).



Vendo Commodore 64 + unidad de Discos + Cartucho de Discos sin estrenar + Cartucho Simon's Basic + juegos en disco (Saucer Attack) + programas de utilidades (Doodle, Hes Mon 64) + manuales correspondientes. Todo ello en perfecto estado (1 año), por 80.000 ptas. (a convenir). Interesados llamar a: Enrique Maler, telf.: (93) 217 87 29.



Deseo intercambiar programas para CBM-64 en *cassette*. Interesados dirigirse a: Sergio Martínez. C/. Condesa de Pardo Bazán, 4, 4.°-2.ª. 08027 Barcelona. Telf.: (93) 340 36 85.



INTERSOFT. Primer Club de intercambio de Software para todos los usuarios del Commodore 64. Intercambio de programas, experiencias, trucos e ideas. Forma parte de un gran Club Commodore 64. Solicita información a: Manuel Arroyo. C/. Maresma, 266, 2.° -2.ª. 08020 Barcelona.



Desearía cambiar toda clase de programas para el C-64 en *cassette* o disco. Dirigirse a: Manuel González Lombardía. C/. Gabriel y Galán, 4, 2.°-A. Peñaranda de Brocamonte (Salamanca).



Desearía intercambiar todo tipo de programas, en cinta o disco, para CBM-64. Dirigirse a: Carlos Veciana Nogués. C/. Mayor, 17. Mora la Nova (Tarragona). Telf.: (977) 40 03 68 (fines de semana).



Desearía intercambiar programas de todo tipo en cinta para el C-64. Enviadme vuestra lista y yo os enviaré la mía. Prometo contestar. Interesados escribir a: Josep M.ª Bosch Vaillo. C/. Joan Maragall, n.° 18. Esparreguera (Barcelona).



Comandos en el C-128

Si aprendemos nuevas formas de carga, grabación y manejo de ficheros ahorraremos tiempo; tiempo que podemos emplear en programar —o jugar— con las instrucciones que vienen implementadas en el nuevo modelo. La mayoría de las nuevas instrucciones para el manejo de la unidad

El BASIC 7.0 del Commodore 128 supone una mejora sustancial sobre las versiones Basic de modelos anteriores. La incorporación de nuevas instrucciones permite una definición más fácil de sprites, efectos sonoros y gráficos en alta resolución sin tener que recurrir a sentencias POKE. En el presente artículo nos centraremos en la utilización de nuevas instrucciones para el manejo de la Unidad de Disco.

de disco funcionan perfectamente tanto en la unidad 1541 como en la nueva 1571. Tanto si posees una como la otra te indicaremos cómo utilizar los nuevos comandos del Commodore 128.

DOCE FORMAS DE CARGAR

Si deseamos cargar un programa BASIC tenemos cuatro posibilidades:

1. LOAD «Nombre del fichero», 8
2. DLOAD «Nombre del fichero»
3. RUN «Nombre del fichero»
4. Pulsar simultáneamente SHIFT-RUN/STOP

Para programar en código máquina o ficheros binarios:

5. LOAD «Nombre del fichero», 8, 1
6. BLOAD «Nombre del fichero»

de disco

7. BLOAD «Nombre del fichero», B banco, P dirección
8. BOOT «Nombre del fichero»

Desde el monitor de código máquina:

9. L «Nombre del fichero», 8
10. L «Nombre del fichero», 8, dirección

Finalmente, dos formas de arrancar con programas autoejecutables (del tipo CP/M, por ejemplo):

11. BOOT
12. Encender o hacer un *reset* al ordenador con un disco de autocarga en la correspondiente unidad.

CARGA DE PROGRAMAS BASIC

Al igual que ocurre en el BASIC 2.0 el número de periférico por defecto con el comando LOAD es el 1 *(Datassette)*, por ello cuando queramos cargar un programa desde la unidad de disco hemos de incluir el número de periférico correspondiente. El nombre del fichero es un nombre de programa que puede contener hasta dieciséis caracteres entrecomillados. Pero no siempre hemos de recurrir a **LOAD** cuando podemos disponer de **DLOAD** y **RUN.**

DLOAD es un nuevo comando, la D es la inicial de «Disco» y asume, por defecto, como periférico el 8 (unidad de disco) y como unidad el 0. En el caso de disponer de una unidad de disco doble hemos de añadir una coma y **DO** o **D1** para especificar la unidad desde la cual queremos efectuar la carga. Desgraciadamente, al menos mientras escribimos esto, Commodore no se ha decidido a comercializar la doble unidad de disco 1572. De todas formas podemos añadir varias unidades a nuestro equipo. Para acceder a una segunda o tercera unidad, **DLOAD** ha de venir seguido de una coma y **U9, U10,** etc.

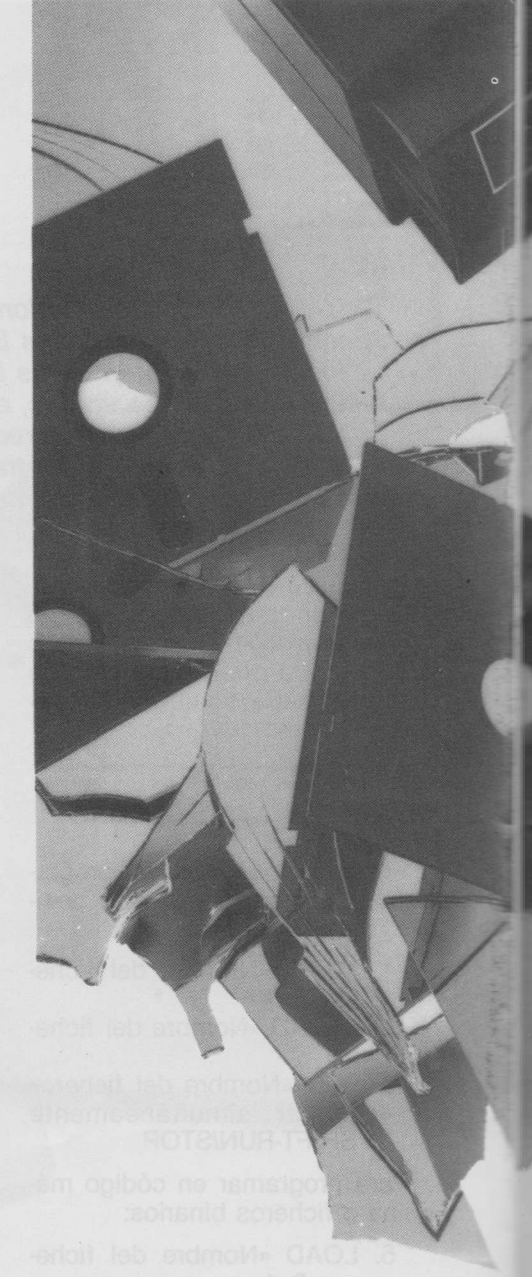
El siguiente comando de la lista, **RUN,** presenta algunos cambios. Todavía se encarga de ejecutar un programa, pero si detrás del comando **RUN** figura un nombre del fichero, éste se carga desde la unidad de disco y se ejecuta directamente. Al igual que **DLOAD** y la mayoría de los demás comandos de disco, podemos especificar el número de la unidad con **D** y el número del periférico con una **U.** El formato quedaría: por tanto, **RUN** «Nombre del programa», D número de la unidad de disco, U número de periférico. **RUN** también lo podemos emplear en el interior de un programa.

En el modo 64 sabemos que pulsando **SHIFT-RUN/STOP** carga y ejecuta el primer programa que se encuentre en la cinta. Pero en el modo 128 esta combinación carga y ejecuta el primer programa del disco.

CARGA DE FICHEROS BINARIOS

Por fichero binario suele entenderse un programa en código máquina, aunque no exclusivamente. También puede tratarse de: redefinición de caracteres, definición de las teclas de función, dibujos en alta resolución, por mencionar algunos ejemplos. Es decir, un fichero binario es un programa, o datos, que ha sido grabado o bien desde el monitor en código máquina o con un comando **BSAVE.**

El comando **BLOAD** carga el fichero especificado por la dirección de inicio. Si estás acostumbrado al VIC o al C-64, reconocerás el formato **LOAD** «nombre del fichero», 8,1. **BLOAD** cumple el mismo cometido, pero no hemos de incluir ni el 8 ni el 1. BLOAD también puede desplazar un programa a otra posición de memoria si le añadimos una B (para el número de banco) y una P (de di-



rección). Con el C-128 sin ampliar, las dos únicas alternativas para el banco son 0 y 1. Los programas BASIC se almacenan en el banco 0, las variables en el banco 1. La dirección de la memoria puede ser cualquiera comprendida entre 0-65535.

BOOT, «Nombre del fichero», carga un fichero binario y ejecuta un **SYS** (llamada y ejecución de una subrutina en código máquina) en la dirección de inicio predefinida. Es el equivalente, en lenguaje máquina, al **RUN** «programa» del BASIC.

También es factible cargar un programa con el monitor de código máquina y el comando L. Tras el nombre del fichero hemos de poner una coma y un 8 (del periférico, unidad de disco). Si deseamos posicionar el programa en otra dirección de memoria, hemos de incluir la nueva dirección.

SECTORES DE AUTOCARGA

Al poner en marcha el C-128, el ordenador comprueba si la unidad de disco se encuentra apagada o encendida. en este último caso intenta la lectura de la pista 1, sector 0 (los 256 bytes de este sector de carga se leen entre las posiciones $ 0BOO - $ 0BFF). Si encuentra las letras «CBM» al inicio de este sector del disco, comienza la autocarga. Podemos comprobarlo de la siguiente manera:

1. Enciende el TV/monitor y la unidad de disco, pero no el 128.
2. Inserta el disco CP/M que acompaña al C-128 en la unidad de disco 1541 o en la 1571.
3. Pon en marcha el ordenador.

El disco de CP/M tiene un sector de autocarga; está concebido para cargar y ejecutar automáticamente el CP/M. Los sectores de autocarga no quedan limitados al CP/M. Es posible crear discos que automáticamente carguen un programa en BASIC o en código máquina. Para crear este tipo de disco aquellos que dispongais de la nueva unidad 1571 podeis cargar el programa **AUTOBOOT MAKER** que viene con la citada unidad. A continuación los tres primeros *bytes* de la pista 1, sector 0 (correspondientes a los caracteres C, B y M), se encuentra el *byte* menor de la dirección de carga, el *byte* más alto, el número de banco para cargar y el número de sectores secuenciales del disco que han de cargarse. Estos 4 *bytes* no son importantes cuando la autocarga se refiere a programas BASIC, por eso habitualmente suelen ser ceros. Al comienzo del octavo *byte* escribiremos el nombre del disco (para el mensaje de carga) y finalizaremos con un cero. Lo siguiente que hemos de poner es el nombre del programa que queramos cargar, igualmente acabado en cero.

UNIR PROGRAMAS

Los ordenadores Commodore suelen presentar algunas dificultades en el proceso de carga y ejecución de un programa en el interior de otro, es decir, problemas al encadenar o unir programas. Las dificultades partían de la forma de almacenamiento de las variables en memoria: el comienzo del almacenamiento de las variables era inmediato al final de los programas en BASIC.

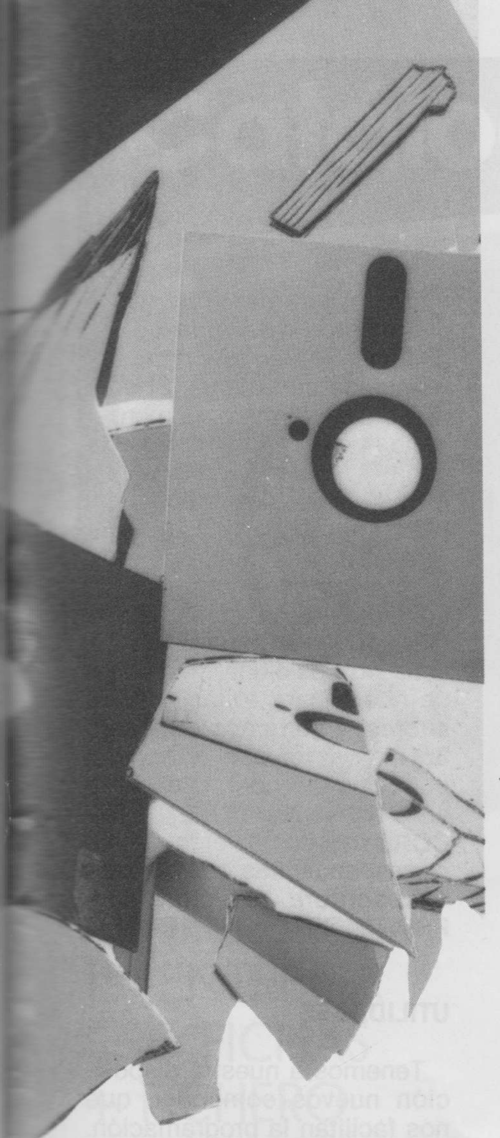
Unir programas es sencillo en el C-128. Puesto que el programa se mantiene separado de las variables no necesitamos preocuparnos por la longitud de los mismos. Para cargar y ejecutar otro programa sigue estas normas:

1. Si quieres conservar las variables del primer programa utiliza **DLOAD.** El segundo programa se carga y ejecuta y las variables se mantienen.
2. Si deseas limpiar las variables utiliza **RUN,** «nombre del programa» donde éste sea el nombre del segundo programa.
3. Para cargar un fichero binario utiliza, o bien **BLOAD,** «nombre del fichero», o **BOOT,** «nombre del fichero».

DE FORMA RAPIDA

Existe una forma rápida para cargar (DLOAD) o ejecutar (RUN) un programa que haya sido grabado de una manera peculiar. Incluye esta línea al principio del programa con el que estés trabajando:

1 REM DSAVE «01 Nombre del programa SHIFT - Barra espaciadora : Esto último significa que mientras mantienes pulsada la te-



cla **SHIFT** pulses la barra espaciadora. Juega con la longitud y colocación de la línea de tal forma que pulsando **TAB** una vez sitúes el cursor frente a **DSAVE** y pulsando dos veces coloques el cursor frente al **1** anterior al nombre del programa. Cuando desees una grabación perfecta de un programa que no esté completo, **LIST 1** y **TAB** dos veces. Cambia el número **01** a **02** y pulsa **RETURN**. Ahora sube el cursor al comienzo de la línea y pulsa **TAB** una vez. Pulsa la tecla **ESC** (próxima a TAB) y la tecla **P.** Esto borra todo aquello que se encuentre entre el cursor y el comienzo de la nueva línea. Pulsa **RETURN**, tu programa quedará grabado en disco con el nuevo número. Más tarde, cuando vuelvas a trabajar con el programa pulsa **F3** para ver el directorio (si va demasiado rápido, la tecla **COMMODORE** lo ralentiza; la tecla **NO SCROLL** lo detiene momentánemente). Cuando lo hayas visto pulsa **STOP.** Sitúa el cursor en el nombre del programa y teclea **DLOAD** o **RUN.** Mejor aún, pulsa **F2 (DLOAD)** o **F6** (RUN). El **SHIFT**--Barra espaciadora de la línea uno coloca unas comillas entre el nombre del programa y los dos puntos. Sin los dos puntos, **DLOAD** o **RUN** interpretarían **NOM** como parte del comando. De la forma aquí descrita la grabación y recuperación de un programa con el que estés trabajando es fácil, cómoda y rápida.

GRABANDO

Algunas formas de grabar programas son:

1. SAVE «Nombre del fichero», 8.
2. DSAVE «Nombre del fichero».
3. BSAVE «Nombre del fichero», B banco, P inicio TO P final.
4. Desde el monitor de código máquina: S «Nombre del fichero», 8, comienzo, final + 1.

Los dos primeros, **SAVE** y **DSAVE,**son los procedimientos habituales para grabar en BASIC. La utilización de **BSAVE** y el manejo del monitor son algo más interesantes. Ambas nos permiten grabar un fichero binario desde las direcciones de memoria que especifiquemos. Hacer notar que cuando se trabaja con el monitor en código máquina hemos de añadir un **1** a la dirección final. Puede parecer que estas dos últimas formas de grabación son más útiles con programas en código máquina. Son muy útiles, por supuesto, pero existen algunas áreas de memoria que podemos grabar (BSAVE) para utilizarlas en un programa BASIC:

$ 0E00 - $ 0FFF	Definición de sprites.
$ 1000 - $ 10FF	Definición de las teclas de función.
$ 1C00 - $ 3FFF	Pantalla en alta resolución.

Las direcciones están en hexadecimal. Para transformarlas en valores decimales podemos utilizar la función **DEC** (Por ejemplo, PRINT DEC («0E00»). Si has creado algunos *sprites* para un juego con **SPRDEF**, puedes grabarlos (BSAVE) en un disco. En el juego tienes entonces que cargarlo (BLOAD) de nuevo en memoria. Esto supone ganar velocidad frente al uso de sentencias **POKE.**

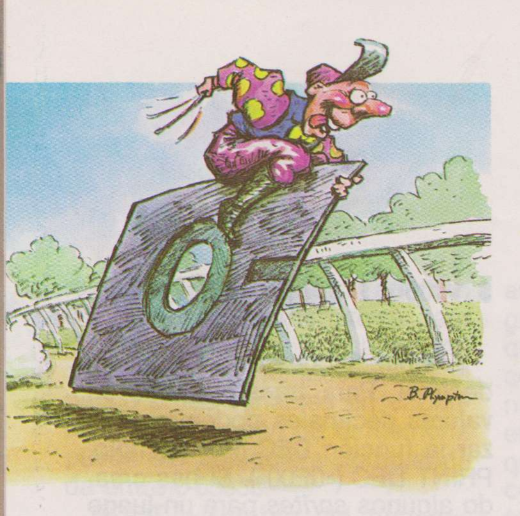
Antes mencionábamos las direcciones de las teclas definibles. Las teclas que podemos redefinir son 10. Son las ocho teclas de función, que podemos definir mediante el comando **KEY** (las teclas van desde **F1** a **F8**), pero también **SHIFT--RUN/STO** y HELP. Si acudimos al monitor en código de máquina y comprobamos la dirección $1000 - $10FF, podemos ver la definición de las 10 teclas. Los primeros 10 *bytes* son la longitud de cada tecla de función. El resto son los caracteres actuales que aparecen al pulsar alguna de ellas. Después de redefinir las teclas, podemos guardar los nuevos valores con **BSAVE.** Para recuperar las teclas definidas hemos de utilizar **BLOAD.**

MANEJO DE FICHEROS SECUENCIALES

DOPEN y **DCLOSE** son dos de las nuevas formas de trabajar con un fichero secuencial. No hay mucho que explicar de ellos; si ya has trabajado con este tipo de ficheros lo entenderás sin ninguna dificultad. La sintaxis de la instrucción quedaría así:

OPEN 3, 8, 4, «Nombre del fichero, S, W».
DOPEN # 3, «Nombre del fiche ro»,W.

Como podemos comprobar en ambos ejemplos, **DOPEN** no pre-



cisa tanta información como **OPEN**. **OPEN es un comando de propósito general, permite al ordenador el acceso a ficheros en periféricos como el** *datassette*, la unidad de disco, una impresora, etc. Es decir, abrimos un canal de comunicación que une cualquier unidad periférica con la unidad central del ordenador. **DOPEN** abre un fichero, en disco únicamente, para operaciones de lectura/escritura. Es por ello que **OPEN** precisa el número de periférico y el canal, mientras que **DOPEN** no. La **S** tras el nombre del programa indica que se va a abrir un fichero secuencial. Hemos de hacer notar también que en el segundo ejemplo la **W** (de escritura, write) ha de situarse fuera de las comillas.

APPEND es una variante de **DOPEN**. Abre un fichero ya existente para una operación de escritura. Cualquier información escrita se añade al final de ese fichero lógico.

FICHEROS RELATIVOS MAS FACILES

Ser capaces de acceder a una determinada información casi aleatoriamente supone un notable ahorro de tiempo. Con los ficheros secuenciales sucede, a veces, que hemos de leer el contenido de 50 fichas para acceder a la 51. Un fichero relativo nos permite acceder a la información que precisamos casi instantáneamente.

En el BASIC 2.0 crear y mantener un fichero relativo requiere mandar un determinado valor utilizando la función CHR$. Si has manejado ficheros relativos en el modo 64, sabrás de su cierta complejidad. Pero no así en el modo 128. En apenas unas cuantas líneas puedes abrir y escribir en un fichero relativo. Un ejemplo, supongamos que deseas 100 fichas con una longitud de 20 caracteres por ficha. Tu programa para crear el fichero vendría a ser algo así:

```
10 DOPEN # 3, «Programa XYZ»,
L20. 20 RECORD#3,100.
30 PRINT#3, «Ultima ficha».
40 RECORD#3,100.
50 CLOSE 3.
```

Y eso es todo. Cuando a **DOPEN** le sigue una **L** y un número, abre un fichero relativo. La longitud de cada ficha viene determinada por **L20.**

Las fichas pueden tener una longitud entre 1 y 254 *bytes*. Puesto que la longitud de la ficha se almacena en el directorio, sólo se precisa utilizar el parámetro **L** al crear el primer fichero.

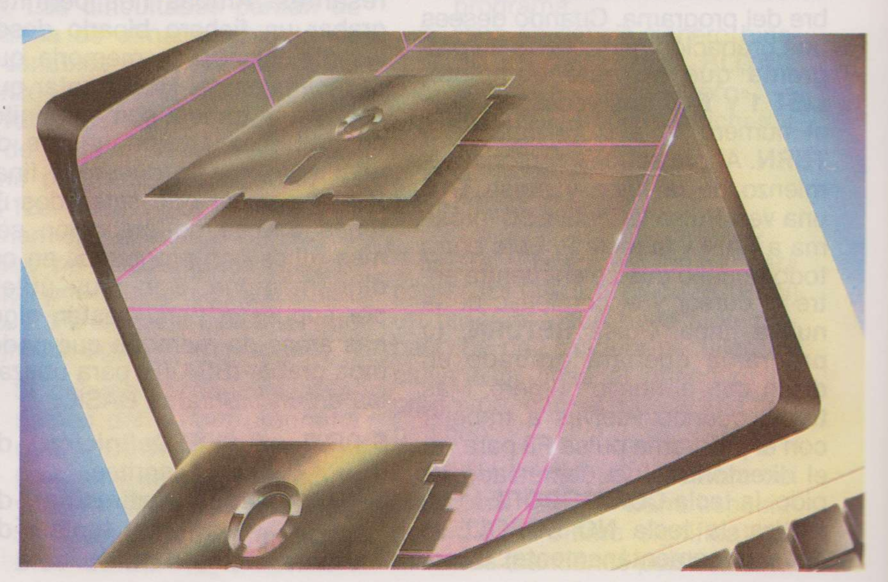
RECORD coloca el puntero en la ficha deseada. Esto es, coloca un puntero para seleccionar cualquier *byte* de cualquier ficha en un fichero relativo. El número de ficha puede oscilar entre 0 y 65535, dependiendo de la longitud de la misma. Podemos además añadir un tercer número si queremos comenzar la lectura o la escritura en una parte determinada de la ficha. Si omitimos este número, comenzaremos por el primer *byte* de la ficha.

En la línea 30 queremos escribir en la ficha 100. La escritura en una ficha que no existe previamente, obliga a la unidad de disco a crear esa ficha y todas las anteriores. La línea 40 posiciona el puntero de nuevo para evitar algunos errores, y cerramos el fichero 3. Una vez creado el fichero, podemos acceder fácilmente a las fichas con **DOPEN** y **RECORD#**. Podemos efectuar las correspondientes operaciones de escritura o lectura con **PRINT#**, GET# o INPUT#.

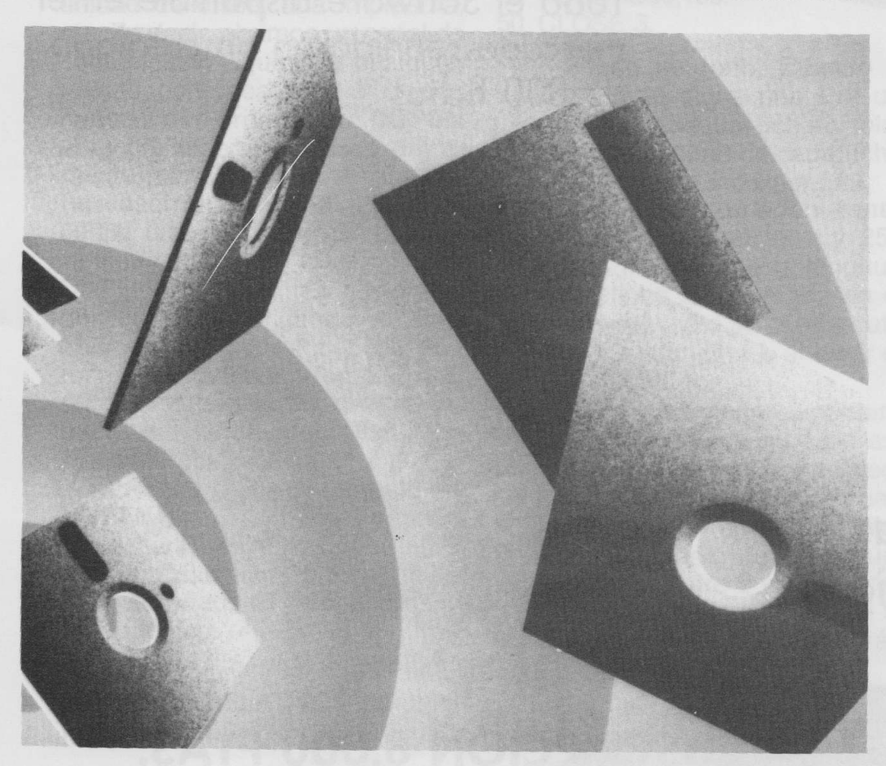
UTILIDADES

Tenemos a nuestra disposición nuevos comandos que nos facilitan la programación. La tecla **F3** está definida para imprimir **DIRECTORY**. Mediante la pulsación de una única tecla podemos ver lo que se encuentra en el disco.

Dos variables reservadas muy utilizadas son **DS** (abre-



viatura de **DISK STATUS**, estado del disco) y **DS$.** La primera devuelve el número de error del disco, y la segunda, imprime el correspondiente mensaje de error. Si la luz roja de la unidad 1541 (la verde en el caso de la nueva 1571) se vuelve intermitente, escribe **PRINT DS$ y podrás comprobar qué es lo que falla. Consulta el manual de tu unidad para el listado completo de los mensajes de error.**



En el interior de un programa DS es mucho más útil que **DS$.** Tras una utilización de un disco, añade una línea **IF DS >19 THEN 500,** donde la línea 500 sea el comienzo de una rutina de depuración de errores. Si no existe ningún error la variable **DS** toma, normalmente, el valor **0**. Pero si **DS** toma un valor igual o superior a 20, significa que algo ha fallado.

Existen algunas excepciones: Error 01 no es propiamente un error, suele producirse tras una operación de borrado **(SCRATCH)**. El mensaje será **FILES SCRATCHED** (ficheros anulados), seguido de una coma y del número de ficheros que han sido borrados. Error 50, **RECORD NOT PRESENT** (No está presente tal ficha), no es motivo de preocupación si has creado o ampliado un fichero relativo. Si escribes en una ficha que no existe, de un fichero relativo, la información se añade al final del disco. La ficha no estaba presente antes de tal operación, de ahí que provoque un error 50.

Algunos comandos facilitan el manejo de los ficheros y lo simplifican. **SCRATCH** borra un fichero del directorio del disco. Tras el comando ha de escribirse el nombre del fichero entre comillas. El nombre del fichero puede contener caracteres comodín (?, *, etc.) cuando queramos borrar algunos ficheros con nombre parecido. Para cambiar el nombre de un fichero **RENAME** «nombre anterior», **TO** «nuevo nombre». El formato es mucho más sencillo que **OPEN 15, 8, 15, «RO:** nuevo nombre-nombre anterior» el formato que requería el VIC o el C = 64.

COLLECT valida el disco. Mediante este comando podemos disponer del espacio en disco que ha sido ocupado por ficheros impropiamente cerrados y para borrar las referencias a esos ficheros en el directorio. Los ficheros impropiamente cerrados aparecen en el directorio con un asterisco; por ejemplo ***PRG.**

DCLEAR cierra todos los canales abiertos en la unidad de disco y cierra todos los ficheros.

CONCAT combina los contenidos de dos ficheros secuenciales. Une el segundo fichero al final del primero, manteniendo el nombre de éste.

CONCAT «Fichero 2», TO «Fichero 1». El fichero resultante es designado fichero 1.

Existen dos comandos pensados para su utilización con unidades de disco dobles. Son **COPY** y **BACKUP**. El primero copia un fichero de una unidad a la otra. Se precisa, pues, una unidad de disco doble. En el caso de una unidad simple, podemos copiar un fichero en el mismo disco, pero teniendo en cuenta que los nombres de estos ficheros han de ser diferentes.

BACKUP copia todos los ficheros del disco origen en el disco destino, en una unidad de disco doble. **BACKUP** copia toda la información del disco, incluyendo el formato. Es decir, podemos utilizar un disco nuevo sin formatearlo previamente. Debido a ello hemos de tener precaución con su manejo, puesto que destruimos toda la información

que hubiera en el disco destino.

PEQUEÑOS INCONVENIENTES

Existen lo que podríamos denominar «pequeñas pegas» en el C-128. Una de ellas es que al pulsar **SHIFT-RUN/STOP** se cargue y ejecute el primer programa del disco. La carga así efectuada estaría muy bien si fuera eso lo que quisiéramos hacer. Pero, más tarde o más temprano, mientras estamos programando podemos pulsar **SHIFT-LOCK** y **RUN/STOP** o la tecla **Commodore** y **RUN/STOP**. Cuando el disco comience a girar, el tiempo del que disponemos para evitar la carga del primer programa es mínimo. Si no conseguimos detenerla, el programa se cargará y ejecutará y habremos perdido aquella parte de programa con la que estuviéramos trabajando. Para evitar este pequeño incordio podemos situar un fichero secuencial al comienzo del disco. Si por error pulsamos las teclas de carga, el ordenador tratará, en vano, de cargar el fichero secuencial. Tomando esta precaución el programa con el que estemos trabajando, estará a salvo.

También podemos, por error, grabar un programa. Las funciones que traen incorporadas **F7** y **F3** son **LIST** y **DIRECTORY,** respectivamente. Muy útiles y cómodas cuando queramos comprobar lo que se encuentra en el directorio o en un determinado programa. Pero en medio de ambas nos topamos con **F5**, definida como **DSAVE**. Si tratamos de listar un programa y accidentalmente pulsamos **F5** y **F7**, el ordenador imprimirá **DSAVE «LIST** y comenzará a grabar el programa con tal nombre.

Finalmente, algo a tener en cuenta al utilizar comando de disco es que las variables han de ir encerradas entre paréntesis. El siguiente ejemplo nos muestra la forma correcta e incorrecta de utilizar las variables:

RENAME H$ TO «Final»: REM INCORRECTO.
RENAME (H$) TO «Reciente»: REM CORRECTO. □

Ficheros secuenciales con el datassette

¿Quién no ha tenido alguna vez que recoger datos tratados a posteriori? Precisamente, esta labor es a menudo una de las más complicadas en la programación y por eso hemos decidido comenzar desde el principio y trabajando con el Datassette.

Vamos a explicar brevemente la administración de ficheros secuenciales para el Commodore 64. Primero tendré que hablar sobre los conceptos básicos de este área, continuando con un ejemplo de programa, de tal forma que su comprensión sea absoluta. **Fichero,** en su significado literal, consiste en uno o varios conjuntos de fichas. Estas se pueden hallar fácilmente en las bibliotecas. En cada una está toda la información referente al objeto o persona que interese. Así sabremos en cada momento qué lector está leyendo qué libro o qué libro ha sido prestado. La reunión de estas fichas da lugar al fichero; pues bien, éste, el informático, sigue el mismo proceso, se encuentra constituido por varios registros, donde aparece la información individual, lo mismo que en la ficha.

EL FICHERO INFORMATICO

La gran ventaja de un fichero informático sobre el fichero clásico (que hemos descrito anteriormente) es su enorme flexibilidad. Hay que tener en cuenta ante todo la rapidez de búsqueda y or-



denación de todos los datos y no debemos olvidarnos del ahorro de espacio que supone.

Antes he mencionado **el registro,** éste se puede comparar con la ficha de un gran fichero, refiriéndonos al ejemplo anterior, en donde se encuentran todos los datos que estarían en la ficha subdivididos por campos.

Si he dicho, refiriéndome siempre al ejemplo anterior, que el equivalente de un registro es la ficha, podemos imaginarnos que **el campo** lo es de un dato individual de ésta. La relación entre los tres

se puede representar en el esquema del cuadro 1.

PROGRAMACION DE FICHEROS

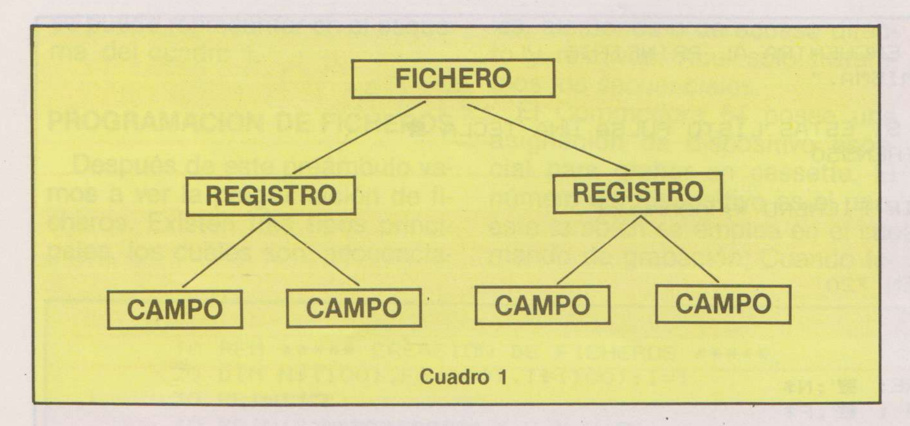
Después de este preámbulo vamos a ver la programación de ficheros. Existen tres tipos principales, los cuales son: secuenciales, aleatorios o de acceso directo y relativos. Aquí sólo trataremos los secuenciales.

El Commodore 64 posee una asignación de dispositivo especial para grabar en cassette. El número del dispositivo es el uno, éste también se emplea en el comando de grabación. Cuando tenemos apretadas las teclas *RECORD* y *PLAY* (del *cassette*), el ordenador graba una cabecera especial en la cinta, que contiene, en el caso de un fichero, el nombre del mismo.

No hay que olvidar que el Commodore 64, al grabar, NO busca el fichero (cuyo nombre ya hemos

```
10 REM ***** CREACION DE FICHEROS *****
20 DIM N$(100),F$(100),T$(100):I=1
30 PRINT"■"
40 PRINT"■■■■■■■■■■■ M E N U ■"
50 PRINT"■■■■■1. GRABAR FICHERO"
60 PRINT"■■■■2. LEER FICHERO"
70 PRINT"■■■■3. INTRODUCIR MAS REGISTROS"
80 PRINT"■■■■4. FINALIZAR"
90 INPUT"■■■■■■■■■■■■■■■■■■ PULSA LA OBCION DESEADA ";O
100 ON O GOTO 110,470,740,1100
110 PRINT"■"
120 PRINT"■■■■■■■■■■ INSTRUCCIONES "
130 PRINT"■■ COJE UNA CINTA Y ASEGURATE DE QUE"
140 PRINT"■ NO TIENES NADA GRABADO EN ELLA"
150 PRINT"■ SI TIENES ALGO GRABADO LO BORRARIA"
160 PRINT"■■ AHORA INTRODUCE LA CINTA"
170 PRINT"■■ CUANDO NO QUIERAS INTRODUCIR MAS DATOS  ESCRIBE <FIN>"
180 PRINT"■■■■■    ■ SI ESTAS LISTO PULSA UNA TECLA ■"
190 GETA$:IFA$=""THEN190
200 PRINT"■"
210 REM ******* ABRIR FICHERO *********
220 OPEN1,1,2,"JUEGOS"
230 PRINT"■"
240 INPUT"■■■ INTRODUCE EL NOMBRE:■ ";N$
250 IF N$="FIN"THEN380
260 INPUT"■■ FECHA CON FORMATO DD/MM/AA:■ ";F$
270 INPUT"■■ TIPO DE PRIGRAMA:■ ";T$
280 INPUT"■■■■■ ESTAS SEGURO S/N:  ";S$
290 IF S$="S"THEN330
300 IF S$="N"THEN230
310 GOTO 280
320 REM *********** GRABAR *************
330 PRINT#1,N$
340 PRINT#1,F$
350 PRINT#1,T$
360 GOTO 230
370 REM ******** CERRAR FICHERO *******
380 CLOSE1
390 PRINT"■"
400 PRINT"■■ PULSA STOP PARA PARAR LA CINTA "
410 FORK=1TO2000:NEXTK
420 INPUT"■■■■■■■■■■■ DESEAS VOLVER AL MENU S/N: ■ ";R$
430 IF R$="S"THEN30
440 IF R$="N"THENEND
450 GOTO 280
460 REM ******** LEER DATOS ***********
470 PRINT"■"
480 PRINT"■■■■■■■■■■■■ INSTRUCCIONES ■"
490 PRINT"■■■  INTRODUCE LA CINTA EN EL CASSETTE "
500 PRINT"■ Y UNA VEZ HECHO ESTO, ASEGURATE DE "
```

```
510 PRINT"■ QUE SE ENCUENTRA AL PRINCIPIO  "
520 PRINT"■ DE LA MISMA."
530 PRINT
540 PRINT"■■■■■■■■ SI ESTAS LISTO PULSA UNA TECLA ■"
550 GETA$:IF A$=""THEN550
560 PRINT"■"
570 REM ****** ABRIR FICHERO **********
580 OPEN1,1,0,"JUEGOS"
590 PRINT"■"
600 IFST AND 64 THEN 720
610 INPUT#1,N$
620 INPUT#1,F$
630 INPUT#1,T$
640 PRINT"■■■ NOMBRE:  ■";N$
650 PRINT"■■■ FECHA :  ■";F$
660 PRINT"■■■ TIPO  :  ■";T$
670 INPUT"■■■■■■■■■ DESEAS VER EL SIGUIENTE (S/N):";S$
680 IF S$="S" THEN590
690 IF S$="N"THEN720
700 GOTO 670
710 REM ******* CERRAR FICHERO ********
720 CLOSE1
730 GOTO 30
740 REM **** PONER MAS REGISTROS ****
750 PRINT"■"
760 PRINT"■■■■■■■■■ PONER MAS REGISTROS ■"
770 PRINT"■■■■  INTRODUCE LA CINTA EN EL CASSETTE "
780 PRINT"■ Y UNA VEZ HECHO ESTO, ASEGURATE DE "
790 PRINT"■ QUE SE ENCUENTRA AL PRINCIPIO  "
800 PRINT"■  DE LA MISMA."
810 PRINT"■■■■■■■■■ SI ESTAS LISTO PULSA UNA TECLA ■"
820 GETA$:IF A$=""THEN550
830 PRINT"■"
840 FORX=1TO1000:NEXTX
850 OPEN1,1,0,"JUEGOS"
860 I=1
870 IF ST AND 64THEN930
880 INPUT#1,N$:N$(I)=N$
890 INPUT#1,F$:F$(I)=F$
900 INPUT#1,T$:T$(I)=T$
910 IF I=100THEN930
920 I=I+1:GOTO870
930 R=I
940 CLOSE1
950 PRINT"■■■■■■■■■REBOBINA LA CINTA HASTA EL PRINTCIPIO"
960 INPUT"■■■■■■ ESTAS LISTO (S/N): ■";R$
970 IF R$<>"S"ANDR$<>"N"THEN960
980 IF R$="N"THEN 960
990 PRINT"■■■■■■■■■■■■■ ESPERA UN MOMENTO"
1000 OPEN1,1,2,"JUEGOS"
1010 FOR I=1TO R
1020 IF ST=64THEN1070
1030 PRINT#1,N$(I)
1040 PRINT#1,F$(I)
1050 PRINT#1,T$(I)
1060 NEXT I
1070 PRINT"■■■■■■■■■■■■■ INTRODUCIR DATOS"
1080 FORX=1TO1000:NEXTX
1090 GOTO 230
1100 REM ************ FIN **************
1110 PRINT"■"
1120 END
```



Cuadro 1

indicado). Esta operación se inicia sin que el ordenador tenga en cuenta el contenido de la cinta desde el punto donde ésta se encuentra. De tal manera que si tenéis algo de información en ese punto de la cinta el ordenador os grabará encima, perdiendo por lo tanto todo lo que tuvierais en ese punto de la cinta. Para evitar esto, yo os aconsejo que utilicéis una sola cinta por fichero, es la manera más segura de saber que no estáis grabando encima de otro.

FICHEROS SECUENCIALES

Como he citado en otras ocasiones, sólo vamos a tratar ficheros secuenciales. Desde un punto de vista etimológico «secuencial» significa «consecutivo». Es decir, que el ordenador va grabando un registro detrás de otro consecutivamente, de tal manera que si quereis ver lo que hay grabado es necesario rebobinar la cinta hasta el principio. Supongamos que tenemos grabado en una cinta musical un disco con varias canciones, si nosotros queremos escuchar una de ellas y no sabemos dónde está tenemos que escucharlas una por una hasta que encontremos la canción que queríamos escuchar; pues lo mismo ocurre con los ficheros secuenciales.

Ahora vamos a crear uno, que va a servir para guardar el nombre de todos nuestros programas y saber en qué fecha los compramos, así como el tipo de programa, juego, tratamiento de texto, etc.

Este fichero contiene diferentes campos: NOMBRE, FECHA, TIPO. En el primero se pone el nombre del programa, en el segundo introducimos la fecha en que éste se compró, y en el último se introduce el tipo de programa de que se trate.

Este programa permite no sólo la creación del fichero, sino también con la opción dos podéis ver los registros que tenéis en él; además os permite mediante la opción tres añadir más registros al fichero. Cuando queráis salir del programa no tenéis más que pulsar la opción cuatro acompañada de la tecla RETURN, con lo cual saldréis del programa y entraréis en modo BASIC.

EXPLICACION DEL PROGRAMA

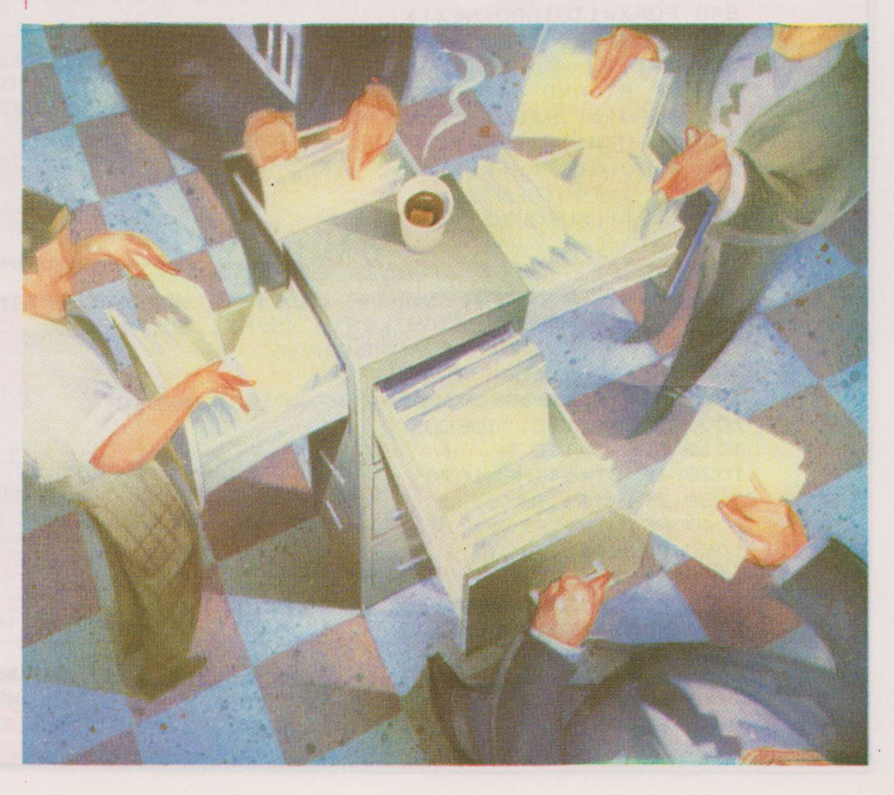
20: Dimensionamos las variables que luego utilizaremos para añadir más registros y ponemos el subíndice de éstas a uno.

40-80: Menú de opciones que dispone el programa.

90-100: Manda al programa a la opción deseada según pulsemos la opción 1, 2, 3, 4.

120-190: Instrucciones para la grabación de los registros.

220 **OPEN 1, 2, «JUEGOS».**
Esta línea es la que nos crea un fichero con el nombre de JUEGOS.



CONCURSO MENSUAL DE COMMODORE MAGAZINE

SU PROGRAMA PARA CUALQUIER SISTEMA COMMODORE PUEDE HACERLE GANAR 5.000 PTAS.

EL PRESENTE CONCURSO ESTA ABIERTO A TODOS NUESTROS LECTORES Y SU PARTICIPACION E INSCRIPCION ES GRATUITA.
LEA LAS BASES DEL CONCURSO

- NO SE ESTABLECEN LIMITACIONES EN CUANTO A EXTENSION, TEMA ELEGIDO O MODELO DE ORDENADOR
- LOS CONCURSANTES DEBERAN ENVIARNOS A LA DIRECCION QUE FIGURA AL PIE, EL CASSETTE O DISKETTE CONTENIENDO EL PROGRAMA, UNA EXPLICACION DEL MISMO Y, AL SER POSIBLE, UN LISTADO EN PAPEL DE IMPRESORA. SE PODRAN ENVIAR TANTOS PROGRAMAS COMO SE DESEE
- LOS PROGRAMAS, PREVIA SELECCION, SERAN PUBLICADOS EN LA REVISTA, OBTENIENDO TODOS ELLOS 5.000 PTAS.
- LA DECISION SOBRE LA PUBLICACION O NO DE UN PROGRAMA CORRESPONDE UNICAMENTE AL JURADO NOMBRADO AL EFECTO POR "COMMODORE MAGAZINE", SIENDO SU FALLO INAPELABLE
- LOS CRITERIOS DE SELECCION SE BASARAN EN LA CREATIVIDAD DEL TEMA ELEGIDO Y LA ORIGINALIDAD Y/O SENCILLEZ EN EL METODO DE PROGRAMACION GLOBAL
- ENVIAR A:
CONCURSO COMMODORE MAGAZINE





240: Esta línea nos permite introducir el primer campo del registro, en este caso NOMBRE.

250 **IF N$ = «FIN» THEN 380.** Esta línea nos determina el final de la escritura de los registros y nos permite volver al menú. Es la línea más importante del programa, pues sin ella no podríamos dejar de grabar registros.

260: Introducimos el segundo campo del registro.

270: Introducimos el tercer campo del registro.

280-310: Estas líneas nos permiten revisar lo que hemos escrito y, en caso de error, escribirlo de nuevo antes de grabarlo en el fichero pulsando la tecla N y después RETURN.

330 **PRINT#1, N$.** Grabamos el primer campo del registro.

340 **PRINT#1, F$.** Grabamos el segundo campo.

350 **PRINT#1, T$.** Grabamos el tercer campo del registro.

360: Volvemos para introducir más registros o para fin.

380 **CLOSE 1.** Cerrarnos el fichero, si omitimos esta instrucción no podríamos volver a acceder a él hasta no haberlo cerrado.

400-450: Instrucciones para volver al menú principal o finalizar.

480-550: Instrucciones para la lectura de registros.

570 **OPEN 1, 0 «JUEGOS».** Abrimos el fichero para lectura de registros.

600 **IF ST AND 64 THEN 710.** Esta sentencia nos determina el «fin» de fichero.

610-630: Leemos los datos del registro.

640-660: Saca por pantalla los datos del registro.

670-700: Nos permite ver más registros o volver al menú principal.

720 **CLOSE 1.** Esta sentencia nos permite cerrar el fichero.

730: Volvemos al menú principal.

760-820: Instrucciones para añadir más registros.

840: Bucle de espera.

850 **OPEN 1, 0, «JUEGOS».** Abrimos el fichero para lectura.

860: Ponemos el subíndice a utilizar a uno.

870-920: Cargamos una tabla con todos los registros que tenemos en el fichero.

930: Hacemos R igual al valor final del subíndice I.

940 **CLOSE 1.** Volvemos a cerrar el fichero.

950-980: Instrucciones para empezar a añadir más registros al final del fichero.

1000 **OPEN 1, 2 «JUEGOS».** Abrimos de nuevo el fichero para grabación.

1010-1060: Grabamos todos los registros que teníamos en el fichero.

1060: Cabecera de aviso para empezar a añadir datos.

1080: Bucle de espera.

1090: Mandamos al programa a introducir más registros.

1110-1120: Finalizamos la sesión y nos salimos del programa. En este momento, para poder ejecutarlo hay que hacer un RUN.

CONCLUSIONES

Espero que este artículo os sirva para entender mejor los ficheros del tipo secuencial.

Tenéis que tener en cuenta que el ordenador dispone de un *buffer* de grabación con capacidad para 250 caracteres, lo que quiere decir que el ordenador realmente no empieza a grabar hasta no tener cubierto el *buffer* o decirle que es fin de fichero, por lo tanto no os asustéis si veis que la cinta no da vueltas.

El programa está preparado para que le podáis añadir más cantidad de campos realizando unas pequeñas modificaciones en las líneas que se encargan de la introducción, grabación y lectura de datos.

EL MITO Y EL RETO



INTERACCION

MOTO JUNIOR
DEL AÑO 1985
Desde los 16 años.

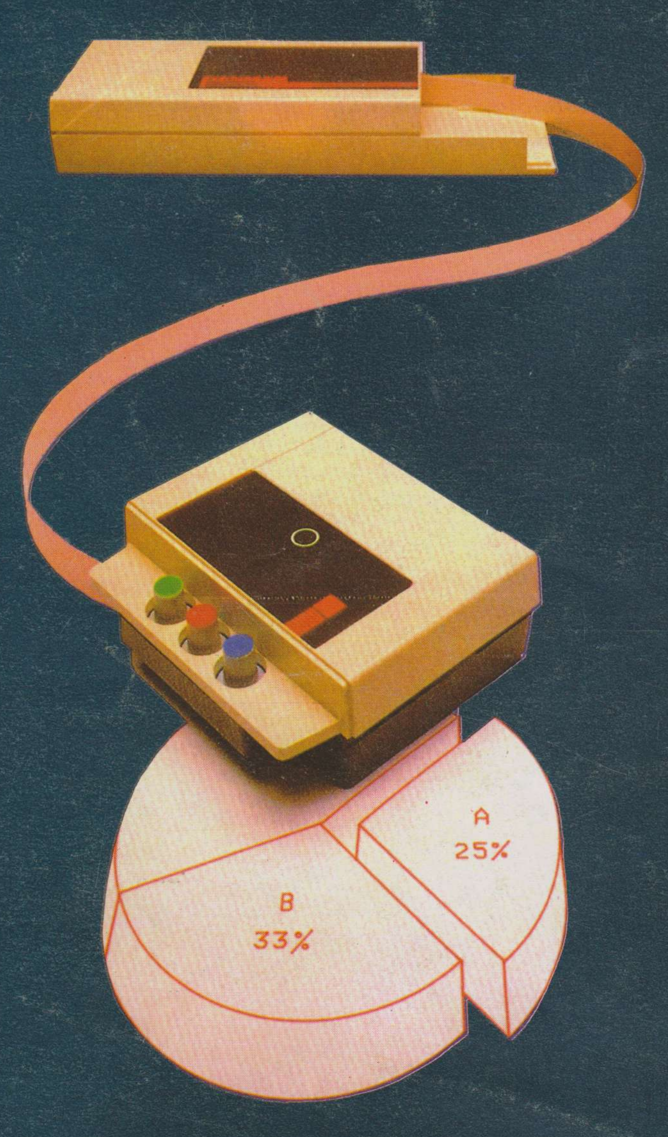
RD-80 Siente la fuerza de la mítica RD. Las siglas con sabor a victoria en mil circuitos. Con la más avanzada tecnología Yamaha. Con el más completo equipamiento. Rápida, suave, segura, bella y silenciosa. Acepta el reto y dale gas: Serás el primero.

MOTUL MOTOR OIL



YAMAHA VA POR DELANTE

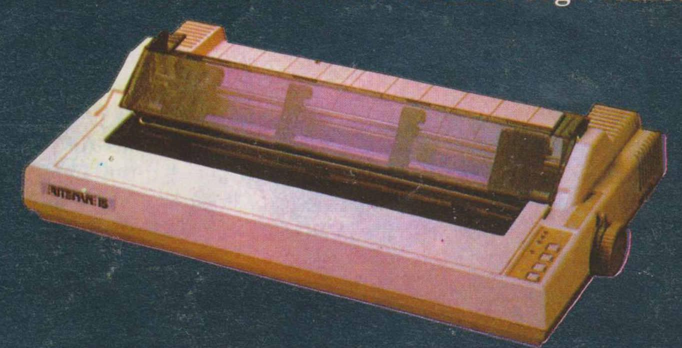
RITEMAN



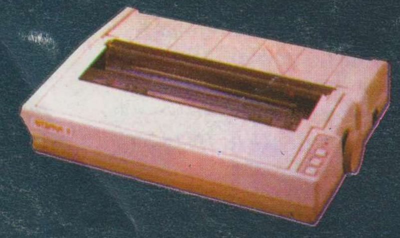


¡NOVEDAD!

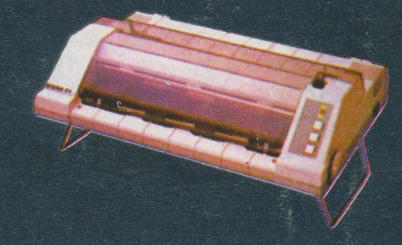
PENMAN: Plotter Robot: 3 colores: 50 mm./seg.: RS232C y RS423



RITEMAN 15 IBM: 160 cps: 8 K buffer: NLQ



RITEMAN 10-II-IBM
160 cps: 8 k buffer: NLQ



RITEMAN F+: CENTRONICS: NLQ: IBM
RITEMAN C+: COMMODORE: NLQ.

DATAMON
DATAMON, S. A.

Provenza, 385
Tel. (93) 207 27 04
Tx: 97791 AEDC
08025 BARCELONA